SIEMENS

SIMATIC

TIA Portal
STEP 7 Basic V10.5

Getting Started

TIA Portal Introduction 1

Simple example 2

Extended example 3

Example "PID control" 4

Example "Motion" 5

12/2009
A5E02651459-01

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠ CAUTION

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation for the specific task, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠ WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be adhered to. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E02651459-01
Ⓟ 12/2009

Copyright © Siemens AG 2009.
Technical data subject to change

Table of contents

1 TIA Portal Introduction 5

1.1 Overview 5

1.2 TIA Portal basics 7
1.2.1 TIA Portal overview 7
1.2.2 Engineering concept 8
1.2.3 Data management 9

1.3 Views in the TIA portal 11
1.3.1 Navigation in the TIA portal 11
1.3.2 Portal view 12
1.3.3 Project view 13

1.4 Example projects 15
1.4.1 Overview 15
1.4.2 Loading a project 17

2 Simple example 21

2.1 Introduction 21

2.2 Create project 21

2.3 Inserting and configuring a PLC 23
2.3.1 Inserting a PLC 23
2.3.2 Overview of device and network editor 26
2.3.3 Configure PLC 29

2.4 Create program 31
2.4.1 What are organization blocks? 31
2.4.2 Open organization block 32
2.4.3 Overview of the program editor 34
2.4.4 What are networks? 35
2.4.5 Inserting LAD instructions 37
2.4.6 What are tags? 42
2.4.7 Defining and interconnecting PLC tags 44

2.5 Test program 48
2.5.1 Loading the program to the target system 48
2.5.2 Test program with program status 54

2.6 Create HMI screen 57
2.6.1 Visualization in the TIA portal 57
2.6.2 Create HMI device with HMI screen 57
2.6.3 What are graphic objects? 65
2.6.4 Creating and configuring graphic objects 66
2.6.4.1 "Machine ON/OFF" button 66
2.6.4.2 Graphic objects "LEDs" 69
2.6.4.3 Graphic object "Conveyor" 74
2.6.4.4 Graphic object "Bottle" with motion simulation 77
2.6.4.5 Modify visibility of motion animation 82

2.7 Testing HMI screen 85
2.7.1 Load HMI screen to the HMI device 85
2.7.2 Simulate runtime 87

3 Extended example 89

3.1 Introduction 89

3.2 Expanding the program 91
3.2.1 Declaring tags in the PLC tag table 91
3.2.2 Programming the conditions for starting the conveyor 95
3.2.2.1 Querying the status of the machine 95
3.2.2.2 Querying the position of the bottle and the status of the heating chamber 97
3.2.2.3 Query pasteurization progress 99
3.2.2.4 Control conveyor 101
3.2.3 Programming the conditions for stopping the conveyor 103
3.2.4 Programming the PLC of the heating 107
3.2.5 Programming the heating period 110
3.2.6 Programming the status light 113

3.3 Test expanded program with program status 117

3.4 Expand HMI screen 123
3.4.1 Graphic object "Heating chamber" 123
3.4.2 Graphic object "Heating chamber LED" 125
3.4.3 Graphic objects "Light barriers" 127

3.5 Simulate HMI screen 132

4 Example "PID control" 135

4.1 Introduction 135

4.2 Set organization block for PID controller 138

4.3 Create technological object PID controller 141

4.4 Load simulation block 143

4.5 Configure PID controller 147

4.6 Changing the heating chamber control 152

4.7 Integrate temperature comparison as condition in control program 154

4.8 Adjust HMI screen 158

4.9 Active PID controller in online mode 161

5 Example "Motion" 165

5.1 Introduction 165

5.2 Insert technological object "Axis" 168

5.3 Configure technological object "Axis" 170

5.4 Enable axis 173

5.5 Position axis relative 176

5.6 Expand HMI screen 180
5.6.1 Change graphic object conveyor 180
5.6.2 Create second graphic object bottle 182
5.6.3 Connect HMI objects with motion instruction 186

5.7 Simulate HMI screen 189

5.8 Start diagnostics view 191

Glossary 197

TIA Portal Introduction

1

1.1 Overview

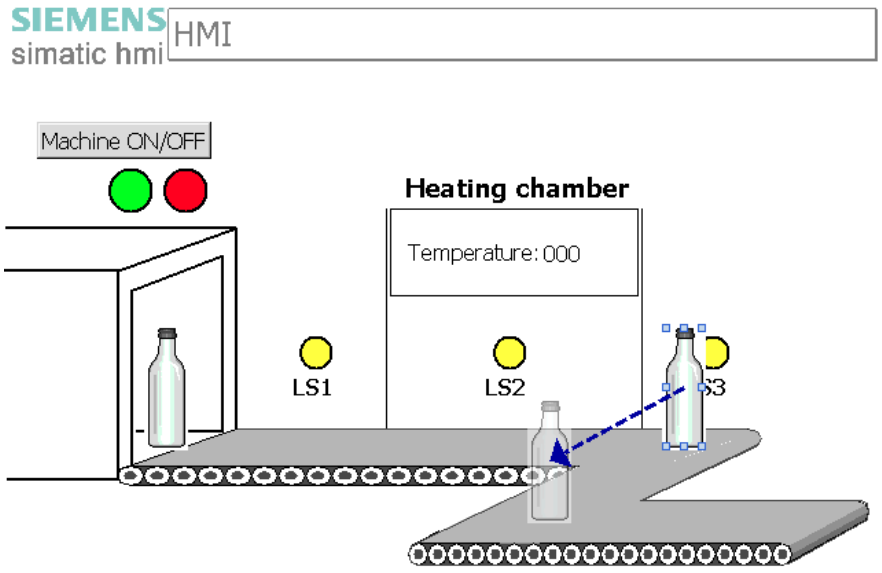
Introduction

Welcome to the "STEP7 Basic V10.5 Getting Started".

Getting Started uses an example project to show you the easy operation of the TIA Portal.

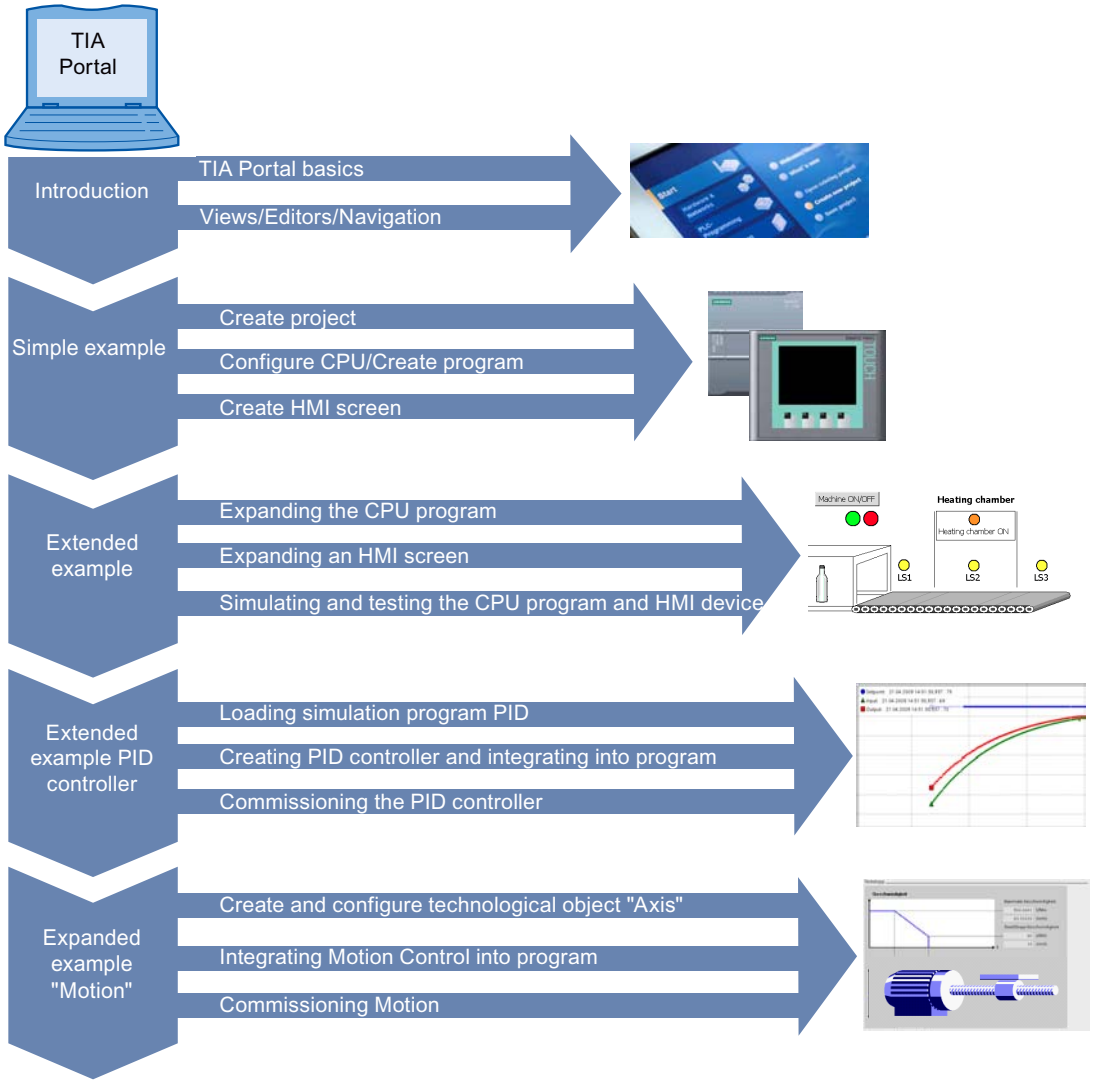
Example project

The example project that is created in this Getting Started is a station for pasteurizing milk in a heating chamber. To illustrate this process in simple terms, the individual bottles are moved on a conveyor belt in the heating chamber and transported to the next stage after completion of the heating process.



Structure of the Getting Started

The example project is expanded with each chapter. Starting from a simple project in which you use only the basic functions of the TIA Portal, you expand the project step by step using increasingly complex functions of the TIA Portal. Experienced users can skip certain chapters; the "Simple Example" chapter offers beginners additional background information on programming and visualization.



1.2 TIA Portal basics

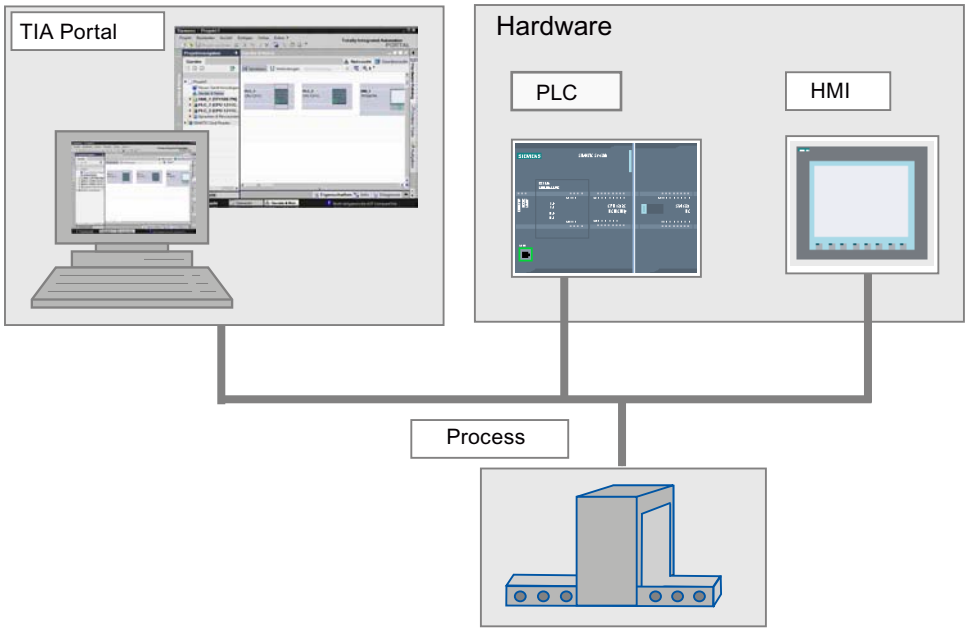
1.2.1 TIA Portal overview

Introduction

The Totally Integrated Automation Portal (TIA portal) integrates various SIMATIC products in a software application with which you can increase your productivity and efficiency. The TIA products work together within the TIA portal and support you in all areas required for the creation of an automation solution.

A typical automation solution encompasses:

- A PLC that controls the process with the aid of the program.
- An HMI device with which you operate and visualize the process.



Tasks

The TIA portal supports you in creating an automation solution. The most important configuration steps are:

- Creating the project
- Configuring the hardware
- Networking the devices
- Programming the PLC
- Configuring the visualization
- Loading the configuration data
- Using the online and diagnostic functions

Benefits

The TIA portal offers the following advantages:

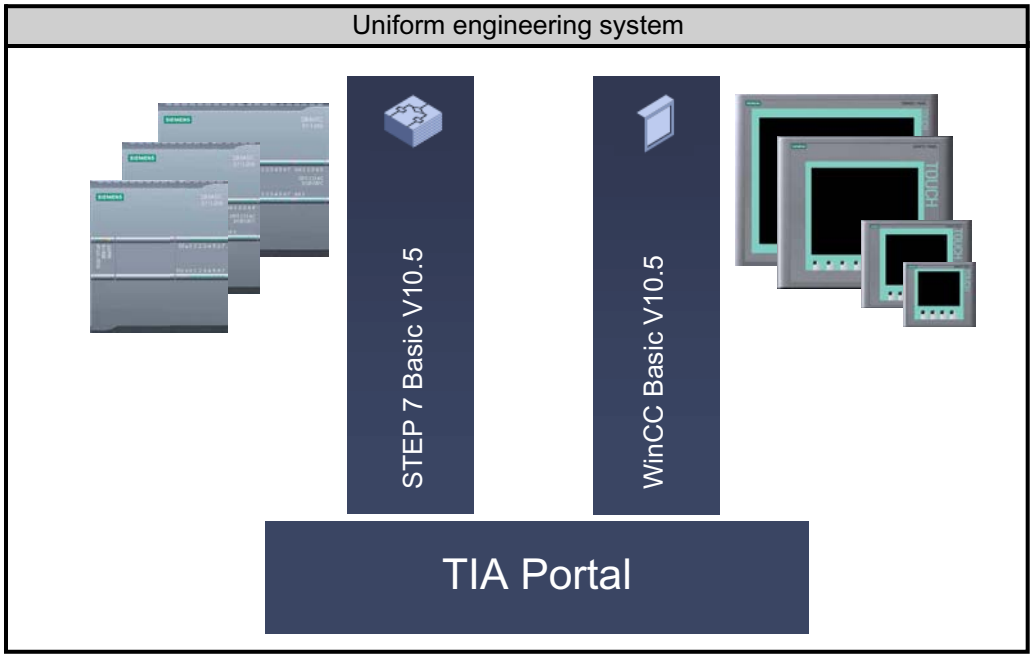
- Common data management
- Easy handling of programs, configuration data and visualization data
- Easy editing using drag-and-drop operation
- Easy loading of data to the devices
- Easy operation
- Graphic supported configured and diagnostics

1.2.2 Engineering concept

Engineering system

You can use the TIA Portal to configure both the PLC and the visualization in a uniform engineering system. All data are stored in one project. The components for programming (STEP 7) and visualization (WinCC) are not separate programs, but rather editors of a system that accesses a common data base. All data are stored in a common project file.

You use a common user interface for all tasks to access all programming and visualization functions at all times.



1.2.3 Data management

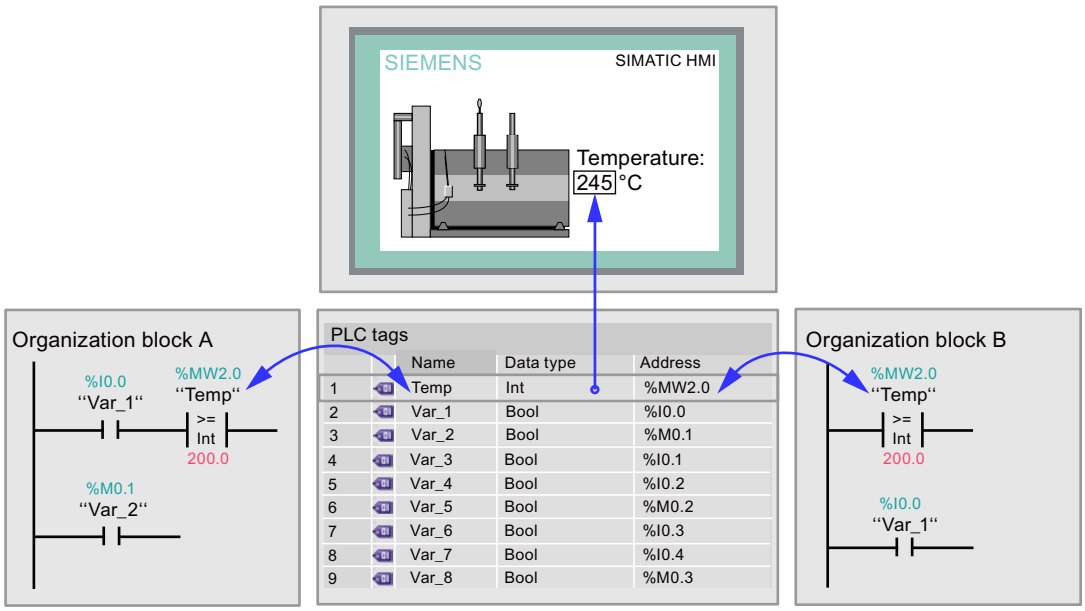
Central data management

In the TIA Portal, all data are stored in one project. Modified application data, such as tags, are automatically updated within the complete project even across several devices.

Symbolic addressing across project parts

If you use a process tag in several blocks of various PLCs and in HMI screens, the tag can be created or modified from any place in the program. In this case it is of no importance in which block of which device you make the modification. The TIA Portal offers the following options for defining PLC tags:

- Definition in the PLC tag table
- Definition in the program editor
- Definition by means of a link with the inputs and outputs of the PLC



All defined PLC tags are listed in the PLC tag table and can be edited there. Modifications are performed centrally and updated continuously. The consistent data management eliminates the need for synchronization between the project participants within a project, for example, between the programmer and the HMI designer.

Library concept

Project parts can be used again via the library both within the project and also in other projects.

- Elements such as blocks, PLC tags, PLC tag tables, interrupts, HMI screens, individual modules or complete stations can be stored both in local and in global libraries.
- Devices and defined functions can be reused.
- The global library allows for an easy exchange of data between projects.

1.3 Views in the TIA portal

1.3.1 Navigation in the TIA portal

Introduction

You will use different views when you create projects. The following section provides you with a preliminary overview of the various views in the TIA portal.

Views of the TIA portal

For your automation projects, the TIA portal offers two different working views that provide fast access to the toolbox and the individual project components:

- Portal view: The portal view supports task-oriented configuration.
- Project view: The project view supports object-oriented configuration.

Navigation

You can use the link in the bottom left corner of the user interface to switch between the portal view and the project view at any time. During the configuration the view is also automatically switched depending on the type of task being executed. For example, if you want to edit an object listed in the portal view, the application switches automatically to the corresponding editor in the project view. When the object has been edited you can switch back to the portal view and continue your work there with the next object or the next activity.

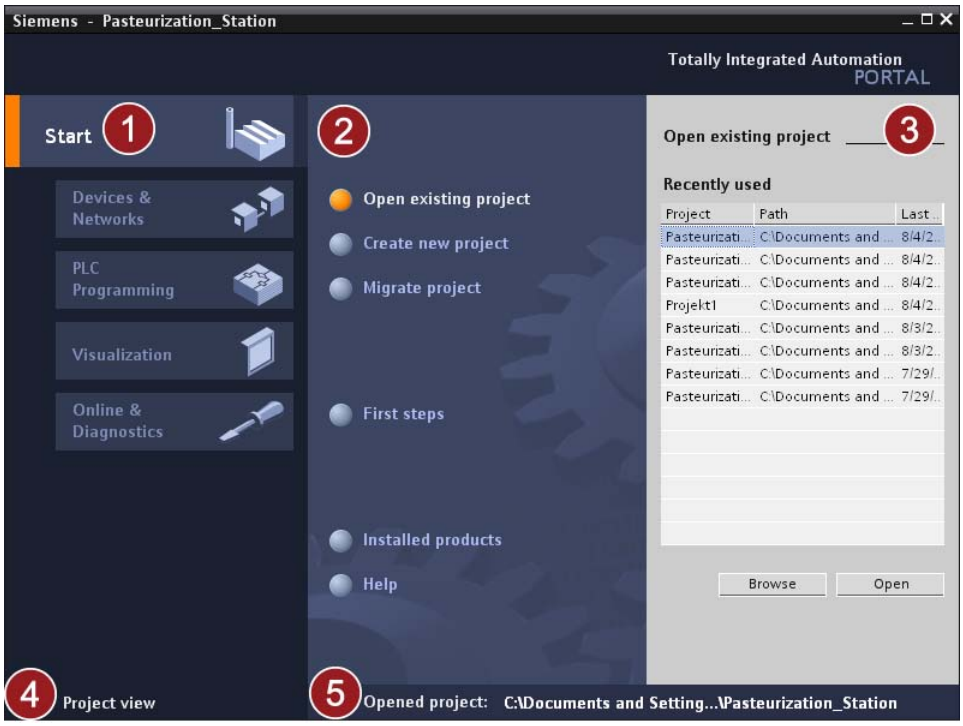
Global saving of project data

When you save a project the complete project is always saved, regardless of which view or editor is open.

1.3.2 Portal view

Portal view

The portal view provides a task-oriented view of the toolbox. The goal of the portal view is to provide a simple navigation in the tasks and data of the project. This means the functions of the application can be reached via individual portals for the most important tasks. The following figure shows the structure of the portal view:



① Portals for the different tasks:
The portals provide the basic functions for the individual task areas. The portals that are provided in the portal view depends on the products that have been installed.

② Actions for the selected portal:
Here, you will find the actions available to you in the portal you have selected. You can call up the help function in every portal on a context-sensitive basis.

③ Selection panel for the selected action:
The selection panel is available in all portals. The content of the panel adapts to your current selection.

④ Switch to project view:
You can use the "Project view" link to switch to the project view.

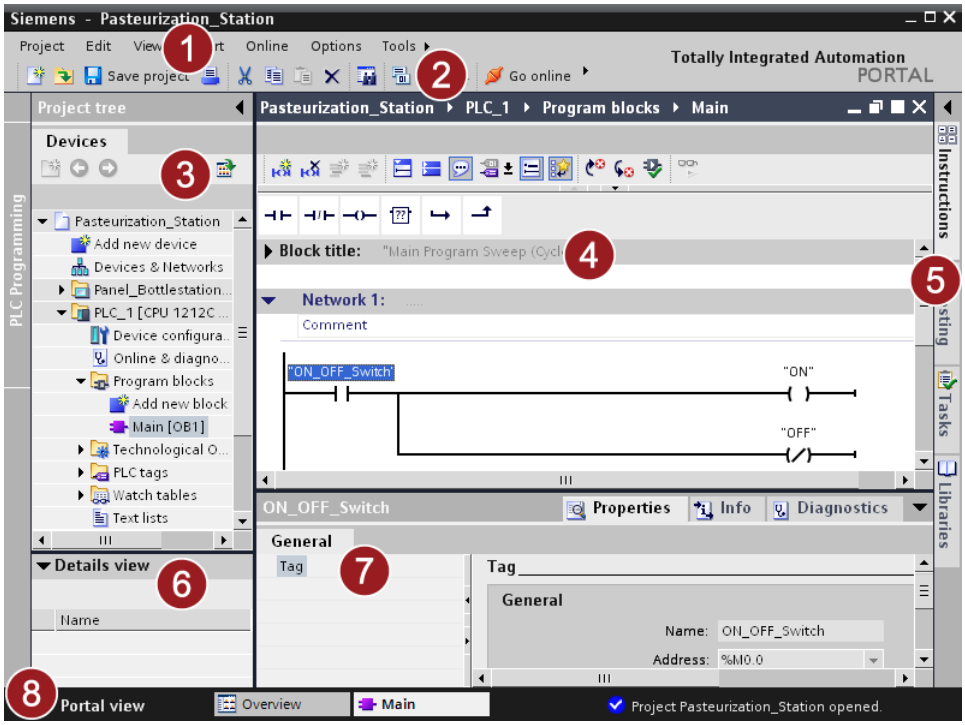
⑤ Display of the project that is currently open:
Here, you can obtain information about which project is currently open.

1.3.3 Project view

Project view

The project view is a structured view of all components of a project. In the project view the various editors are available that you can use to create and edit the corresponding project components.

The following figure shows the structure of the project view:



① Menu bar:
The menu bar contains all the commands that you require for your work.

② Toolbar:
The toolbar provides you with buttons for commands you will use frequently. This gives you faster access to these commands than via the menus.

③ Project tree:
The project tree gives you access to all components and project data. You can perform, for example, the following tasks in the project tree:

- Add new components
- Edit existing components
- Scan and modify the properties of existing components

④ Work area:
The objects that you can open for editing purposes are displayed in the work area.

⑤ Task cards:
Task cards are available depending on the edited or selected object. The task cards available can be found in a bar on the right-hand side of the screen. You can collapse and reopen them at any time.

⑥ Details view:
Certain contents of a selected object are shown in the details view. This might include text lists or tags.

⑦ Inspector window:
Additional information on an object selected or on actions executed are displayed in the Inspector window.

⑧ Switching to portal view:
You can use the "Portal view" link to switch to the portal view.

Note

You open and close individual windows of the project view with the key combination "<Ctrl> + 1-5". You will find an overview of all key combinations in the information system of the TIA Portal.

1.4 Example projects

1.4.1 Overview

Introduction

The Getting Started is divided into four chapters each of which is based on the one before it. The current status at the end of a chapter is saved in a project file. To skip a chapter, you can load the respective status of the preceding chapter.

Contents of the example projects

The following overview describes the target group of the respective chapter. The projects are saved as ZIP files and are available for download at the following address:

http://support.automation.siemens.com/WW/view/en/40263542

Click on "Info" to see the ZIP files.

- Simple example

  These chapters are intended for beginners without prior knowledge. In the easy example you configure the PLC and one HMI device, create a short program and one HMI screen for visualization. At the end of the chapter the machine can be switched on and off using the HMI device.

- Extended example

  If you have prior knowledge you can load the easy example and continue from this point. In this extension of the easy example you create a complete program in LAD and the HMI elements required for visualization of the process.

  The project status after the "Simple Example" chapter is stored in the "Simple_Example.ZIP" file.

- Extended example PID controller

  In this chapter you add the technological object "PID controller" to the extended example. You use the PID controller to automatically control the temperature in the heating chamber.

  If you only want to use the function of the technological object "PID", you can load the project status after the "Extended Example" chapter from the "Extended_Example.ZIP" file.

- Extended Example Motion Control

  In this chapter you add the technological object "Axis" to the extended example. You use the technological object "Axis" to control a second conveyor. You use a motion control instruction to modify the position of the bottle on the conveyor. The project status before this chapter is stored in the "Extended_Example_PID.ZIP" file.

The project status at the end of the complete Getting Started is stored in the "Extended_Example_Motion.ZIP" file.

WARNING

Use the example projects only for test purposes

The supplied examples and descriptions are only intended to provide you with an introduction to the basic functions of the TIA Portal.

- Use the examples only in a test environment and not in a machine that is operational.
- Loading the supplied example programs in a machine that is operational can cause malfunctions, program errors and serious property damage and personal injuries!

1.4.2 Loading a project

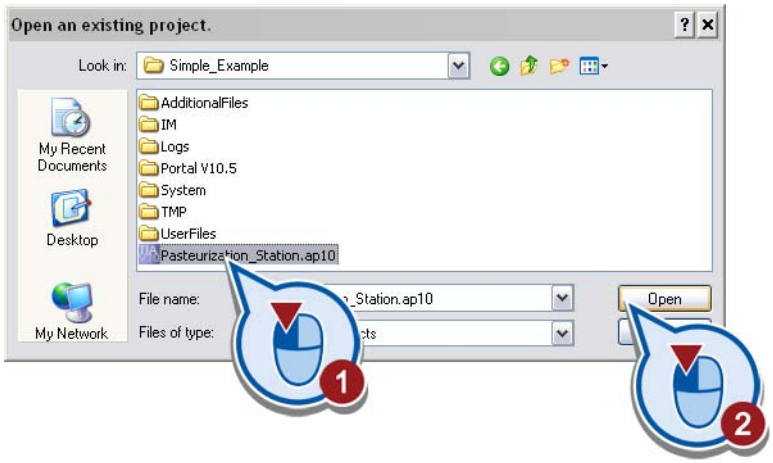
Introduction

The Getting Started starts with the chapter "Simple example (Page 21)". If you do not want to work your way through the Getting Started from the very beginning, the following steps will show you how to load a project status.

Procedure

Proceed as follows to load a project:

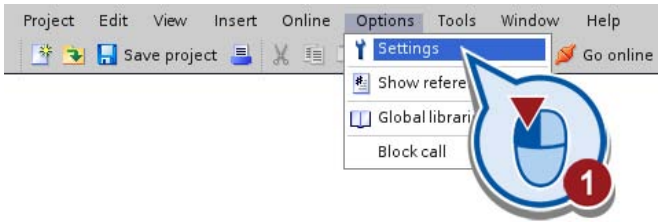
1. Use the portal view of the TIA Portal to open the corresponding project.

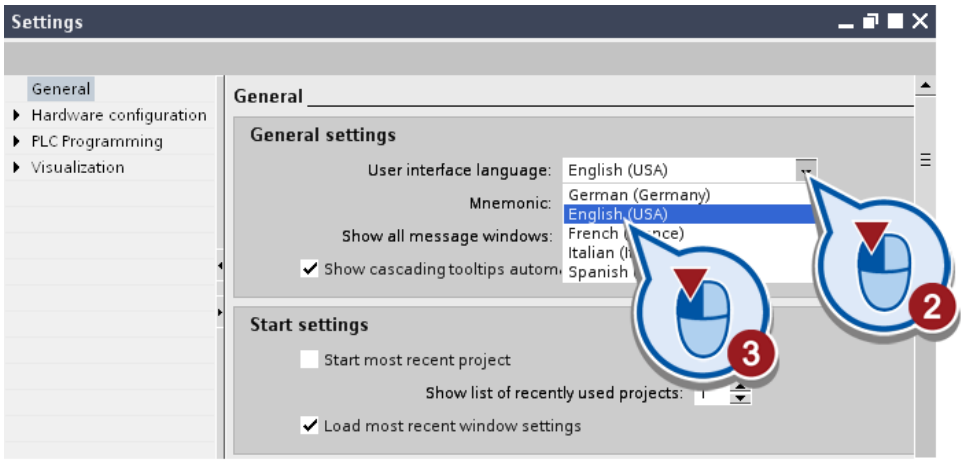


The project is loaded.

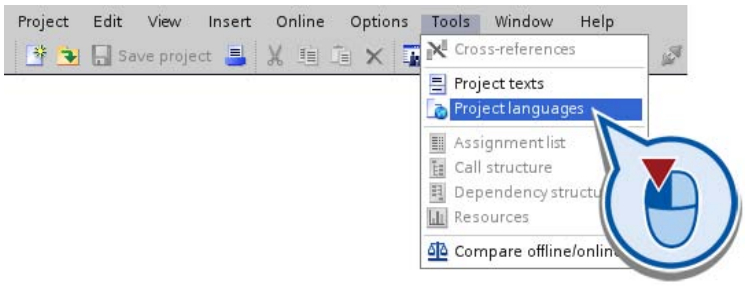
2. Open the project view.

3. Select the language of the user interface.



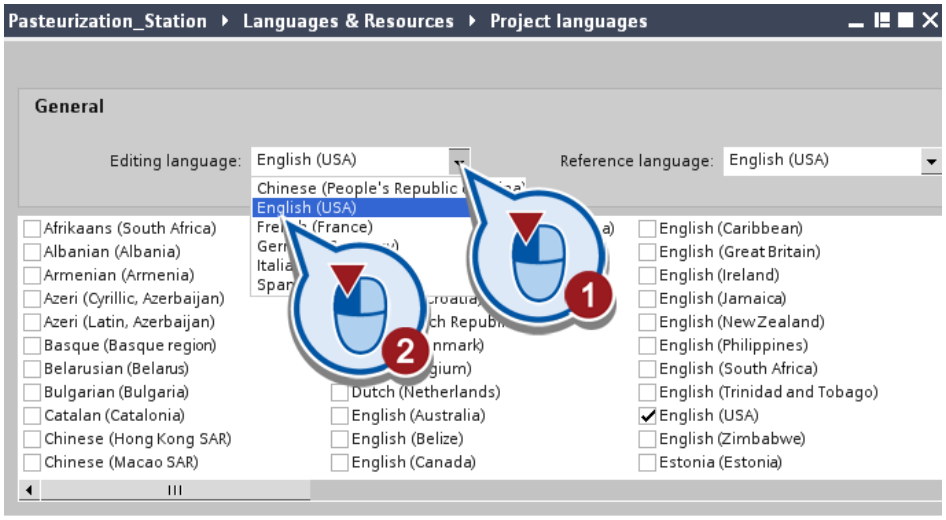


4. Open the window for project language setting.



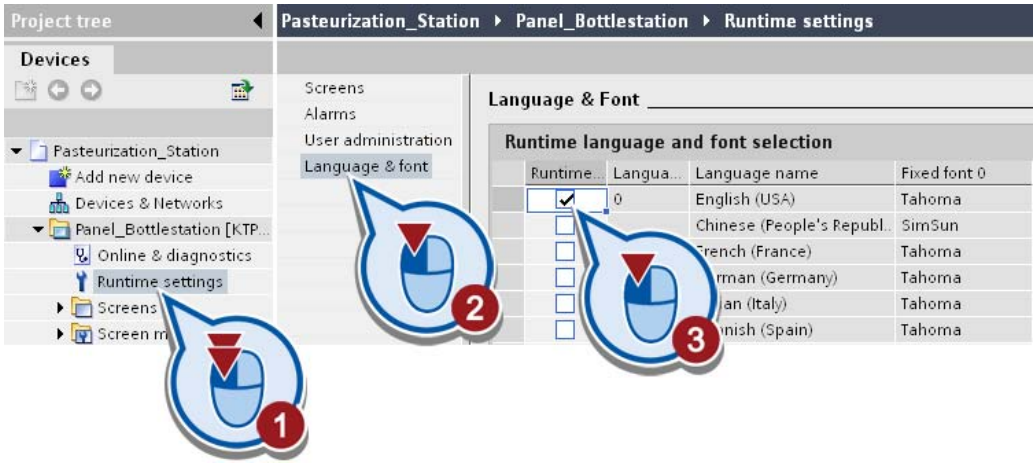
Use the project languages to select the language in which you want to display the project texts, such as text blocks or comments.

5. Select the language for the project texts.



Use the project languages to select the language in which you want to display the project texts, such as text blocks or comments.

6. In the project view, enable the desired language in the runtime settings.



Use the language setting to specify which languages will be loaded in the HMI device. The number of languages that can be loaded in the HMI device will depend on the HMI device that is used. The language with sequential number "0" will be shown first during the start process.

Note

Language switching

Tag names are not affected by project language switching, because the tag name must be unique.

Simple example

2

2.1 Introduction

Steps

In the first part of the Getting Started you program an electric pushbutton for switching a machine on and off. You start the machine by pressing the button once. If you press the button again the power supply is interrupted and the machine switched off.

To do so, follow these steps:

- Create project
- Configure PLC
- Create program
- Load program in the PLC
- Test program
- Create HMI screen

2.2 Create project

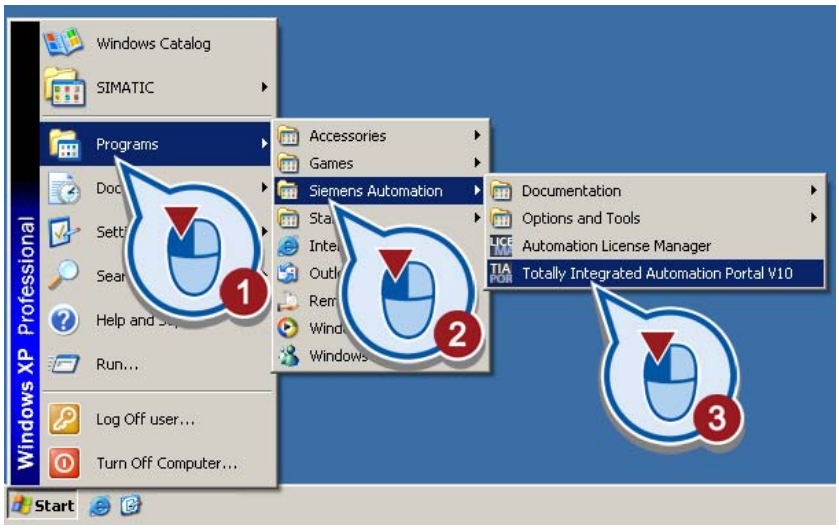
Introduction

The following steps show you how to create a new project. In the project, the data and programs that arise from the creation of an automation task are stored in an orderly manner. For this example, open the Totally Integrated Automation Portal in the portal view. The Start portal contains commands for creating a new project and for opening an existing project.

Procedure

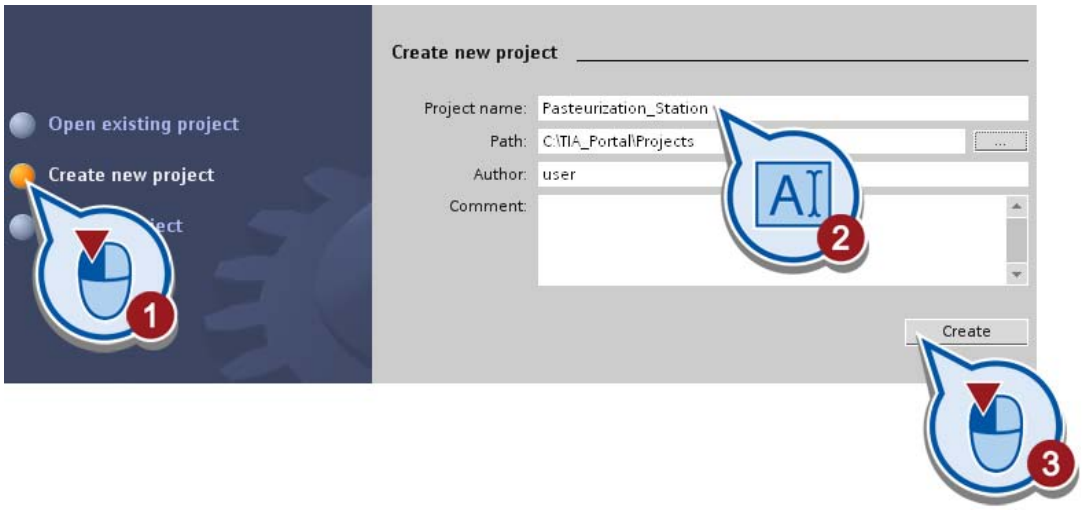
To create a new project, follow these steps:

1. Start the Totally Integrated Automation Portal.



The Totally Integrated Automation Portal opens in portal view.

2. Create the project "Pasteurization_Station" under any path.



Result

You have created a new project. In the next section you will add a new PLC to the project and configure its properties.

2.3 Inserting and configuring a PLC

2.3.1 Inserting a PLC

Introduction

The following steps show you how to insert a PLC in portal view and to open its configuration in the project view. The type of PLC that you create in the project must be compatible with the available hardware.

Requirements

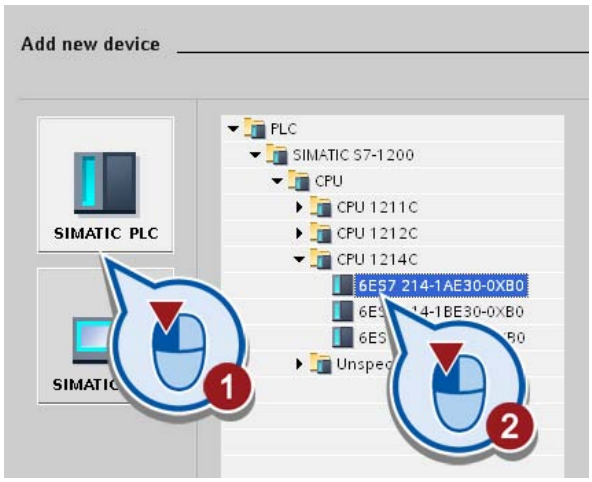
You have created a project.

Procedure

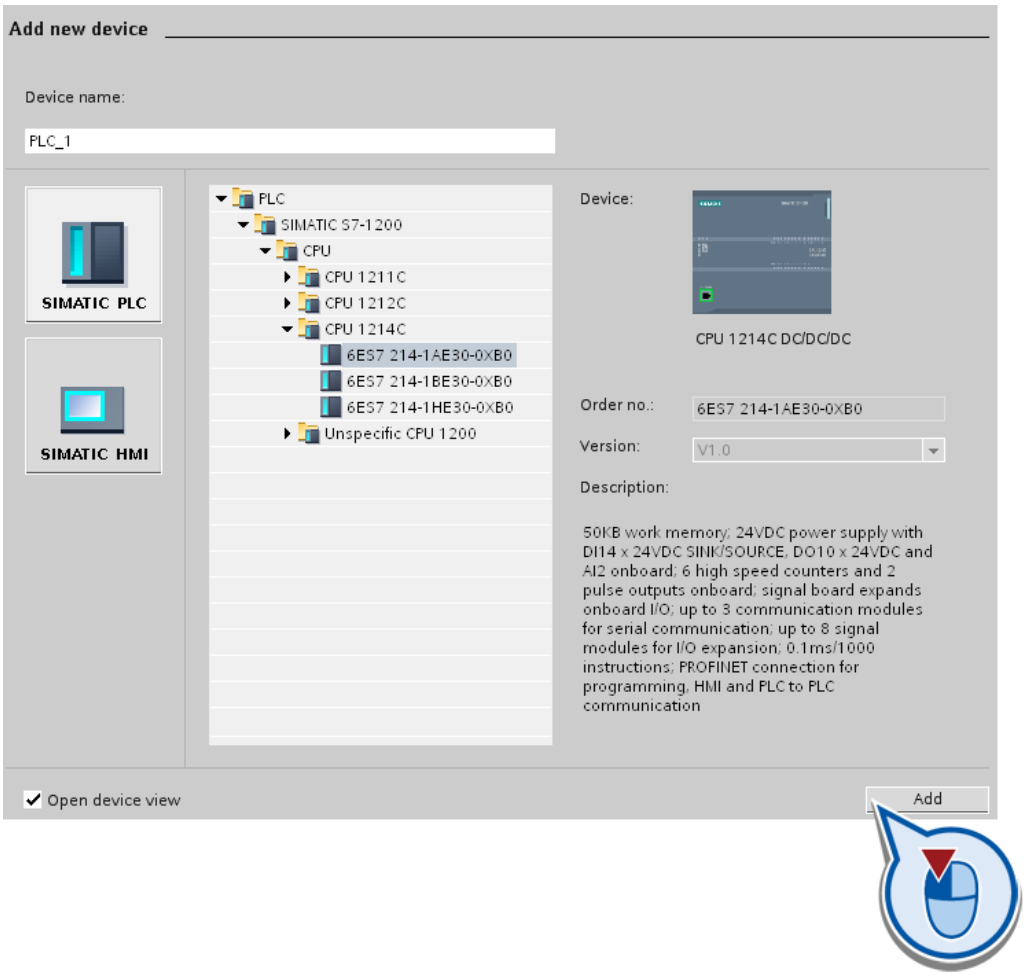
To add a new device to the project, follow these steps:

1. Use the portal to add a new device.

2. Select the desired PLC.

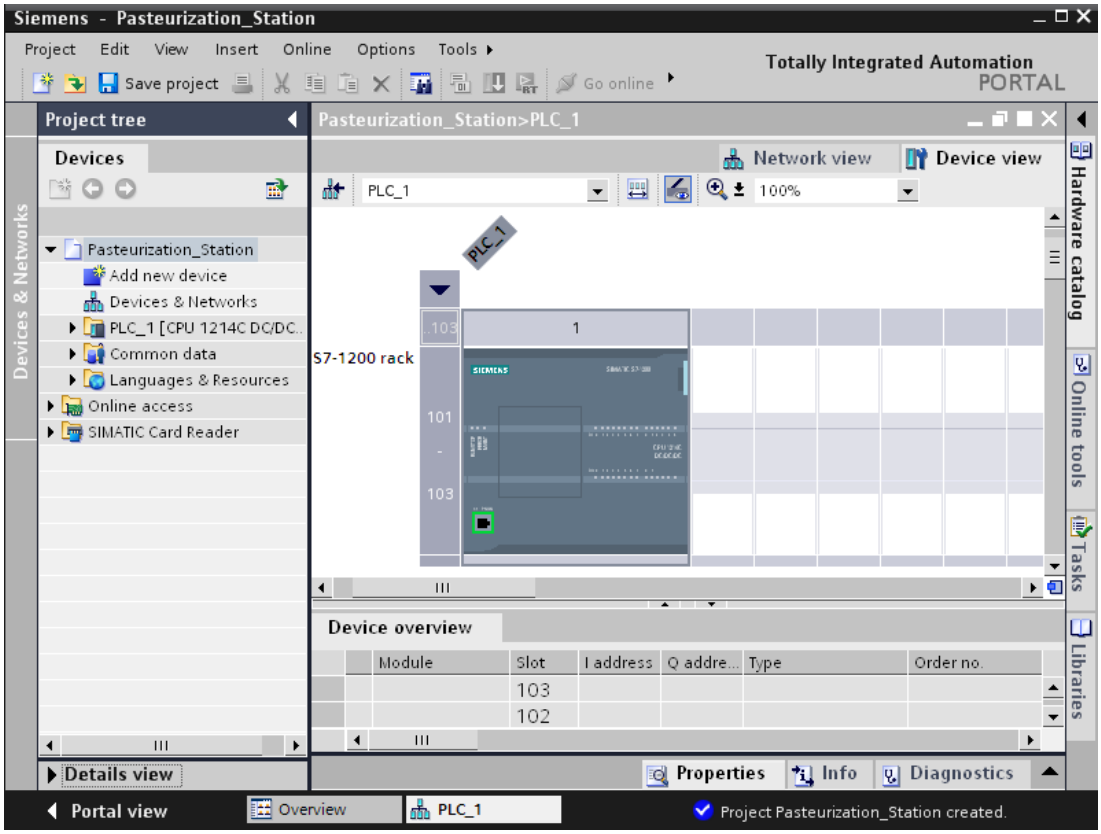


3. Make sure that the "Open device view" option is enabled.
   If the option is not enabled, left-click in the option to enable it.
4. Click "Add".


Add new device
Device name:
PLC_1
SIMATIC PLC
SIMATIC HMI
PLC
SIMATIC S7-1200
CPU
CPU 1211C
CPU 1212C
CPU 1214C
6ES7 214-1AE30-0XB0
6ES7 214-1BE30-0XB0
6ES7 214-1HE30-0XB0
Unspecific CPU 1200
Device:
CPU 1214C DC/DC/DC
Order no.:
6ES7 214-1AE30-0XB0
Version:
V1.0
Description:
50KB work memory; 24VDC power supply with DI14 x 24VDC SINK/SOURCE, DO10 x 24VDC and AI2 onboard; 6 high speed counters and 2 pulse outputs onboard; signal board expands onboard I/O; up to 3 communication modules for serial communication; up to 8 signal modules for I/O expansion; 0.1ms/1000 instructions; PROFINET connection for programming, HMI and PLC to PLC communication
Open device view
Add

Result

You have created a new PLC in the project and have opened the PLC in the device view of the device and network editor.



2.3.2 Overview of device and network editor

Function of the device and network editor

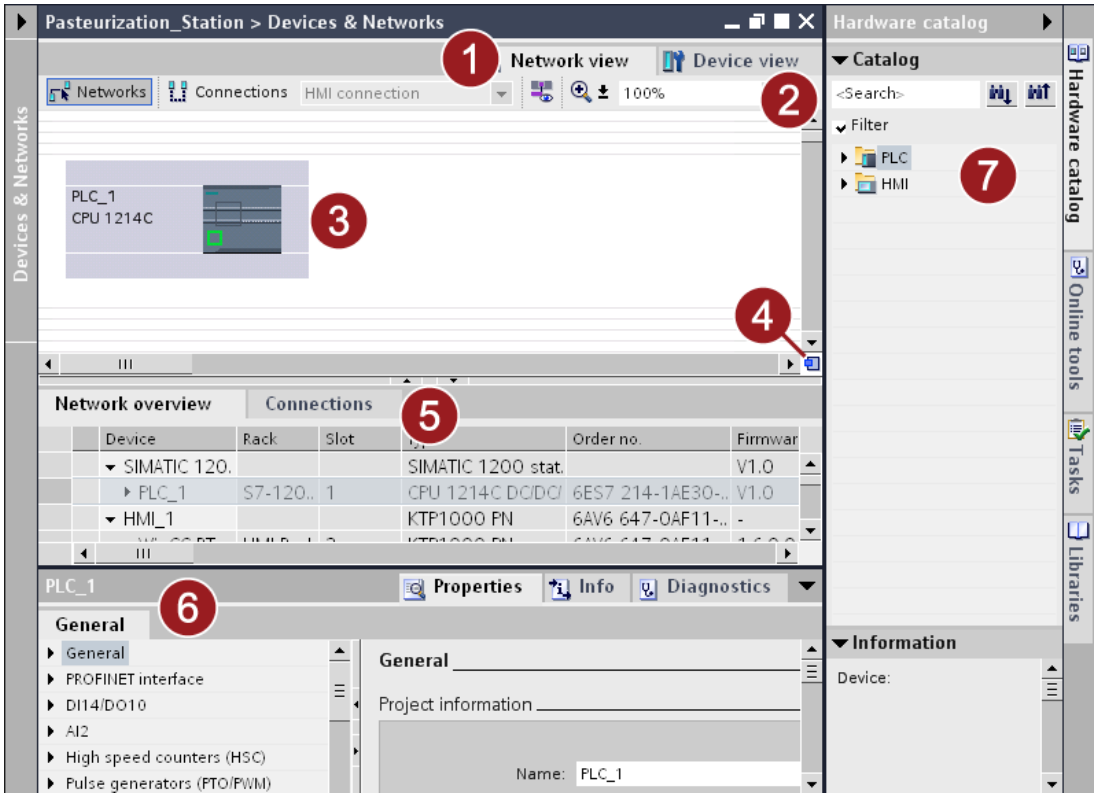
The device and network editor is the integrated development environment for configuring, networking and assigning parameters to devices and modules. It is made up of a network view and a device view. You can switch between these two editors at any time.

Network view

The network view is the work area of the device and network editor in which you can perform the following tasks:

- Configuring and assigning device parameters
- Connecting devices to each other

The following figure shows the structure of the network view:



① Tab for switching between device and network view

② Toolbar:

The toolbar includes the tools for graphic networking of devices, the configuration of connections and the display of address information. Use the zoom function to change the representation in the graphic area.

③ Graphic area:

The graphic area displays network-related devices, networks, connections and relations. In the graphic area you can insert devices from the hardware catalog (7) and connect these to each other by means of available interfaces.

④ Overview navigation:

The overview navigation provides an overview of the objects created in the graphics area. By holding down the mouse button, you can quickly navigate to the desired objects and display them in the graphic area.

⑤ Table area:

The table area provides an overview of the devices, connections and communication connections in use.

⑥ Inspector window:

The Inspector window shows information about the currently selected objects. You can edit the settings of the selected objects in the "Properties" tab of the Inspector window.

⑦ "Hardware catalog" task card:

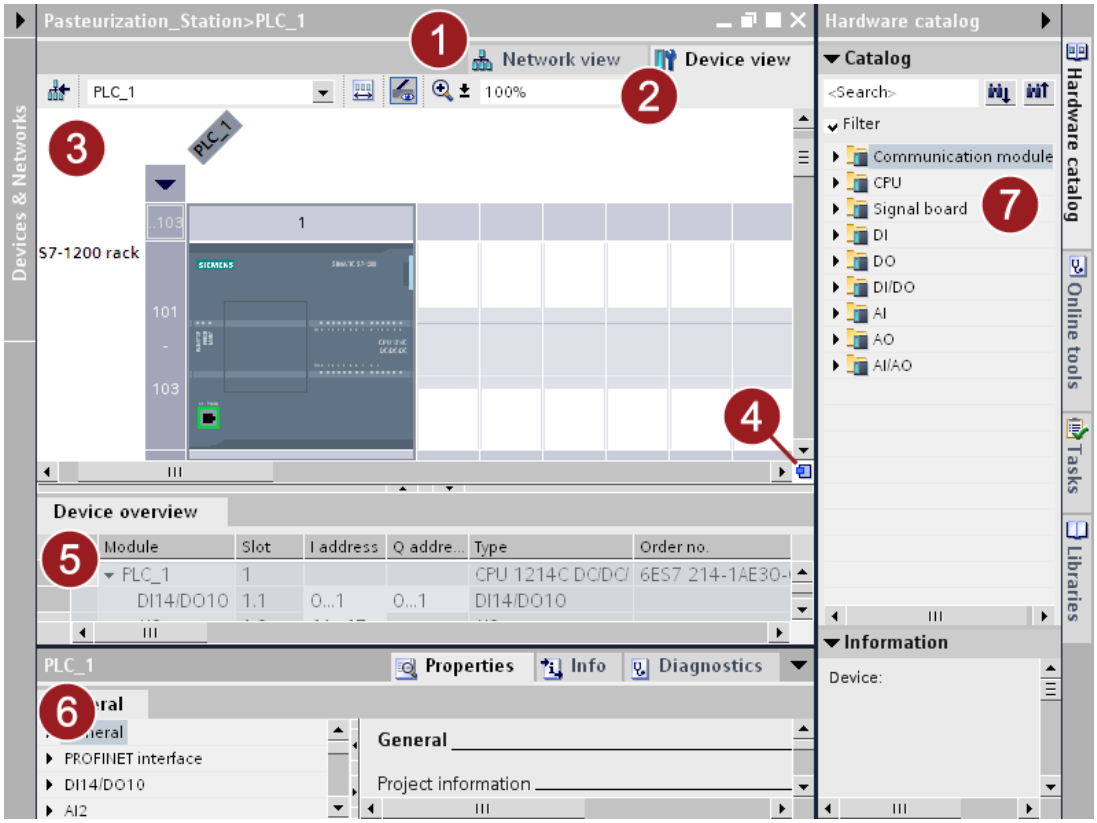
The "Hardware catalog" task card gives you easy access to various hardware components. Drag the devices and modules you need for your automation task from the hardware catalog to the graphic area of the network view.

Device view

The device view is the work area of the device and network editor in which you can perform the following tasks:

- Configuring and assigning device parameters
- Configuring and assigning module parameters

The following figure shows the structure of the device view:



① Tab for switching between device and network view

② Toolbar:

You can use the toolbar to switch between the various devices and to show and hide certain information. Use the zoom function to change the representation in the graphic area.

③ Graphic area:

The graphic area of the device view displays devices and associated modules that are assigned to each other via one or more racks. In the graphic area you can drag additional hardware objects from the hardware catalog (7) into the slots on the racks and configure these.

④ Overview navigation:

The overview navigation provides an overview of the objects created in the graphics area. By holding down the mouse button, you can quickly navigate to the desired objects and display them in the graphic area.

⑤ Table area:

The table area gives you an overview of the modules used and the most important technical and organizational data.

⑥ Inspector window:

The Inspector window shows information about the currently selected objects. You can edit the settings of the selected objects in the "Properties" tab of the Inspector window.

⑦ "Hardware catalog" task card:

The "Hardware catalog" task card gives you easy access to various hardware components. Drag the devices and modules you need for your automation task from the hardware catalog to the graphic area of the device view.

2.3.3 Configure PLC

Introduction

The following steps show you how to configure of the PROFINET interface of the inserted PLC.

Requirements

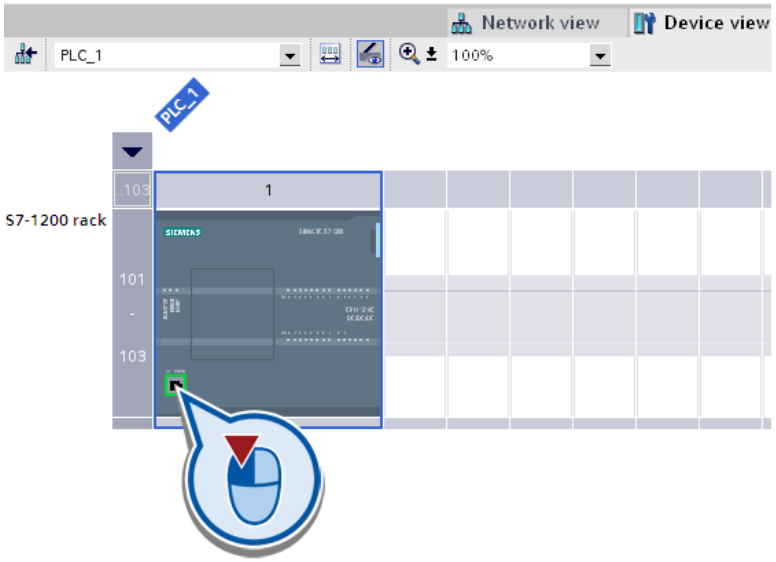
- The project has been created.
- The PLC has been opened in the device view of the device and network editor.

Procedure

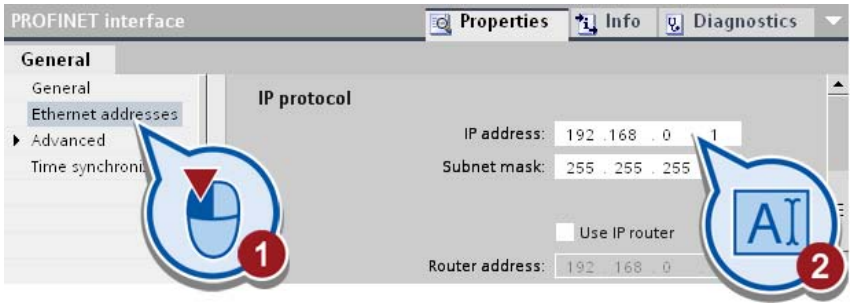
To configure the PLC, follow these steps:

1. Select the PROFINET interface in the graphic view.

   The properties of the PROFINET interface are displayed in the Inspector window.



2. Enter the IP address of your PLC under "Ethernet addresses" in the Inspector window.



3. Save the project by clicking on the "Save project" icon on the toolbar.
4. Close the device and network editor.

Result

You have configured the PLC by setting the properties of the PROFINET interface.

2.4 Create program

Introduction

With the PLC, the organization block "Main [OB1]" is automatically created in the project. In the following section you create the user program in the organization block.

2.4.1 What are organization blocks?

User program

A user program can consist of one or several blocks. You will have to use at least one organization block. The blocks include all functions that are necessary for processing your specific automation task.

The program tasks include:

- Processing process data, e.g. linking binary signals, reading in and evaluating analog values, defining binary signals for output, and outputting analog values
- Reaction to interrupts, for example, diagnostic error interrupt if the measuring range of an analog expansion module is overshot
- Error handling in normal program execution

Organization blocks

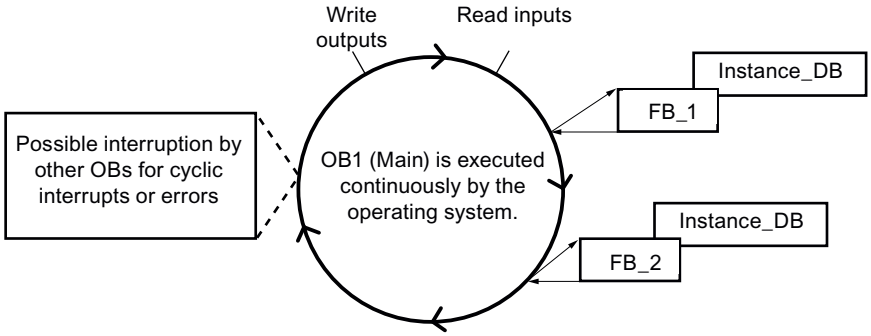
Organization blocks (OBs) form the interface between the operating system of the PLC and the user program. They are called by the operating system and control the following operations:

- Startup behavior of the automation system
- Cyclic program execution
- Interrupt-driven program execution
- Error handling

At least one program cycle OB must be present in an automation project. The program that determines the behavior of the PLC is written in this program cycle OB. The operating system calls the OB once in each cycle and thereby starts the execution of the program contained in the OB. The cycle restarts after the end of each program execution.

The program execution of an organization block can be interrupted by calling other organization blocks. During complex automatic tasks the program is structured and divided into several blocks that are called in the program cycle OB and executed one after the other.

The following figure shows the execution of a program cycle OB:



A program cycle OB with the name "Main [OB1]" is automatically created when you insert a PLC into the project. In this organization block you create the program of the Getting Started project.

2.4.2 Open organization block

Introduction

The following steps show you how to open the organization block in the program editor. The program editor is the integrated development environment for creating the program.

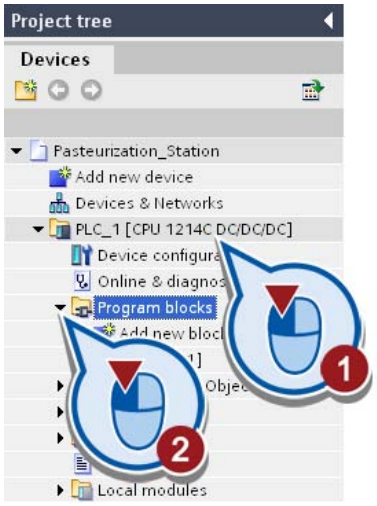
Requirements

A PLC is present in the project.

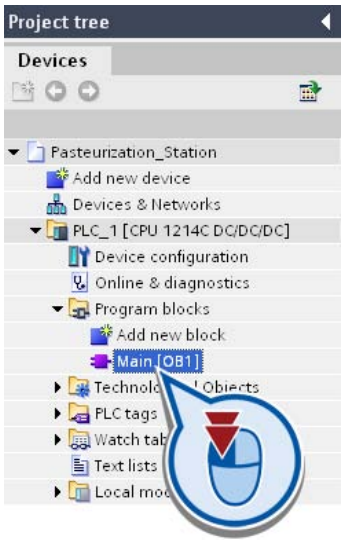
Procedure

To open the organization block "Main [OB1]", follow these steps:

1. Open the "Program blocks" folder in the project tree.

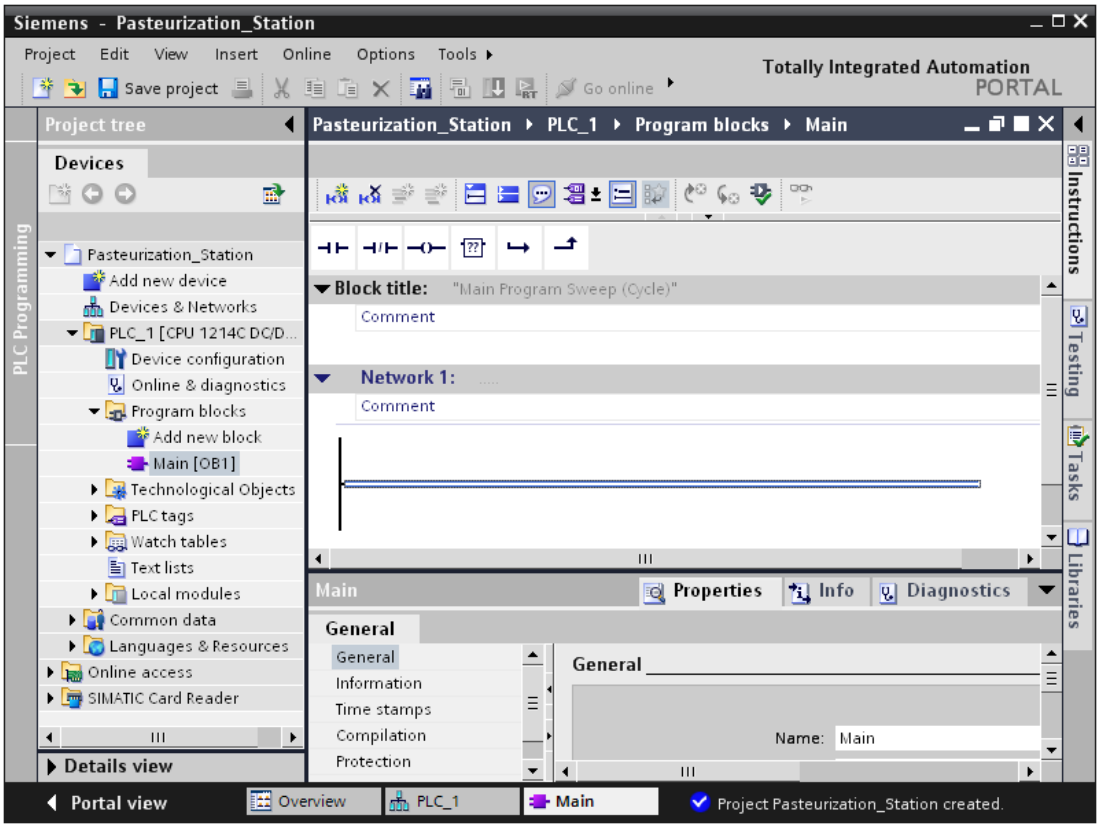


2. Open the organization block "Main [OB1]".



Result

You have opened the organization block "Main [OB1]" in the program editor and you can create your program here.



Note

Adjusting the work area

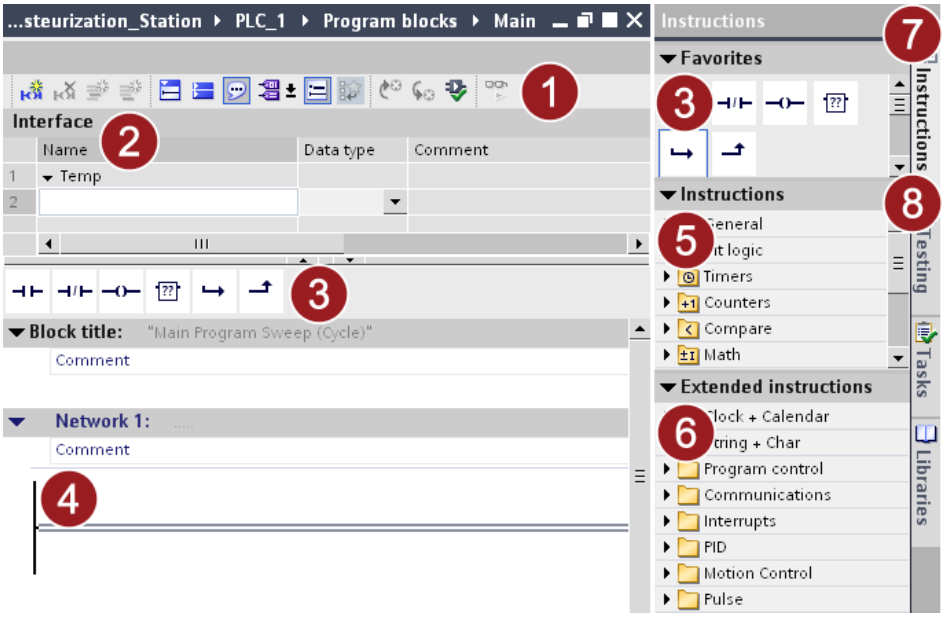
You can move the window of the work area, dock off, and divide it horizontally or vertically. More information on this topic is available in the information system of the TIA Portal.

2.4.3 Overview of the program editor

Function of the program editor

You can use the program editor to create the blocks of the program. The program editor consists of several areas which, depending on function, support the execution of the various programming tasks.

The following figure shows the structure of the program editor:



① Toolbar:

The toolbar allows you access the principal functions of the program editor, such as:

- Insert, delete, open, and close networks
- Show and hide absolute operands
- Show and hide network comments
- Showing and hiding favorites
- Show and hide program status

② Block interface:

The block interface allows local tags to be created and managed.

③ "Favorites" pane in the "Instructions" task card and favorites in the program editor:

The favorites provide quick access to the instructions that you use frequently. The "Favorites" pane can be individually extended with additional instructions.

④ Instruction window:

The instruction window is the work area of the program editor. You can perform the following tasks here:

- Creating and managing Networks (Page 35)
- Entering titles and comments for blocks and networks
- Inserting instructions and supplying them with tags.

⑤ "Instructions" pane in the "Instructions" task card

⑥ "Extended Instructions" pane in the "Instructions" task card

⑦ "Instructions" task card

The "Instructions" task card contains the instructions you use to create the contents of the program.

⑧ "Testing" task card

2.4.4 What are networks?

Introduction

The program of an organization block is divided into networks. The networks are used to structure programs. Each block can contain up to 999 networks.

A network is automatically created in the organization block "Main [OB1]".

Networks in the LAD programming language

The program of an organization block can be created with different programming languages. For the example project, the organization block "Main [OB1]" is edited with the graphic programming language LAD.

The representation of this programming language is based on circuit diagrams, i.e. each LAD program of a block is divided into networks, each of which consists of one power rail and at least one rung.

A network can be extended by adding additional rungs. You can use branches to program parallel connections in the specific rungs. Rungs and networks are executed from top to bottom and from left to right.

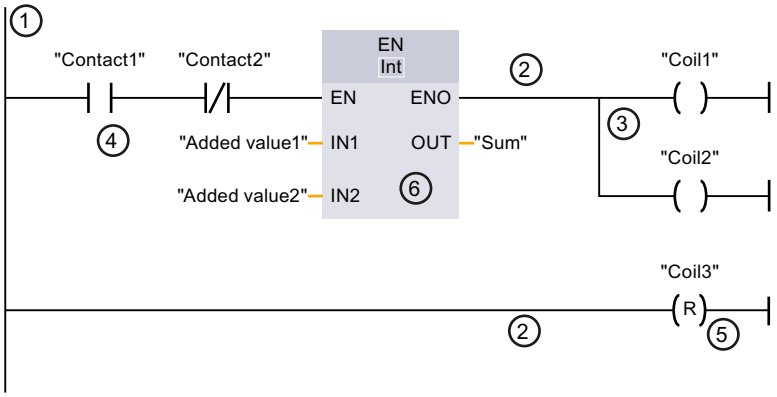
LAD instructions

You create the actual contents of the program using the LAD instructions that are available in the "Instructions" task card of the user interface. There are three different types of LAD instructions:

- Contacts: You can use contacts to create or interrupt a current-carrying connection between two elements. In such cases the elements can be the LAD program elements or the edges of the power rail. The current is relayed from left to right. You can use contacts to query the signal state or the value of an operand and control it depending on the result of the current flow.
- Coils: You use the coils to modify binary operands. Coils can set or reset a binary operand depending on the signal state of the result of logic operation.
- Boxes: Boxes are LAD elements with complex functions. The empty box is an exception. You can use the empty box as a placeholder in which you can select the required operation.

In the "Instructions" task card you can find various variations of contacts, coils and boxes that are sorted into various folders depending on their function. Most LAD instructions must be supplied with tags.

The following figure shows an example of a programmed LAD network:



① Power rail
② Rung
③ Branch
④ Contact
⑤ Coil
⑥ Box

2.4.5 Inserting LAD instructions

Introduction

The following steps show you how to create the program for switching on and off the example machine in the first network of the organization block "Main [OB1]". To do this, insert a branch and the following instructions:

- Normally open contact
- Output coil
- Negated coil

In addition you use a branch to program a parallel connection in the "LAD" programming language.

Normally open contact

The following figure shows the icon of the normally open contact in the program:

<Operand>

---| |---

The activation of the normally open contact depends on the signal state of the associated tag (<operand>).

- If the tag has signal state "1", the normally open contact is closed. Power flows from the left power rail through the normally open contact into the right power rail and the signal state at the output of the instruction is set to "1".
- If the tag has signal state "0", the normally open contact is not activated. The power flow to the right power rail is interrupted and the signal state at the output of the instruction is reset to "0".

In the course of this project you will also use a normally closed contact (Page 95) which essentially has the same function as the normally open contact, but responds to the signal state of the tag in the opposite way.

Output coil

The following figure shows the icon of the "Output coil" instruction in the program:

<Operand>

---( )---

You can use the "Output coil" operation to set the bit of a specified tag (<operand>). If the signal state is "1" at the coil input, the tag bit is set to "1". If the signal state is "0" at the coil input, the tag bit also supplies the signal state "0".

Negated coil

The following figure shows the icon of the "Negated coil" instruction in the program:

<Operand>

---(/)---

The "Negated coil" operation inverts the signal state and assigns the corresponding bit to the specified tag (<operand>). If the signal state is "1" at the coil input, the bit is reset to "0". If the signal state is "0" at the coil input, the bit of the tag is set to "1".

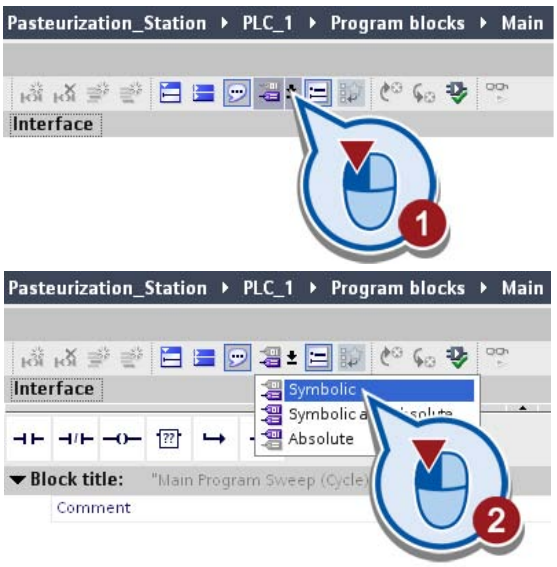
Requirements

- The project has been created.
- The PLC is configured.
- The organization block "Main [OB1]" is open.

Procedure

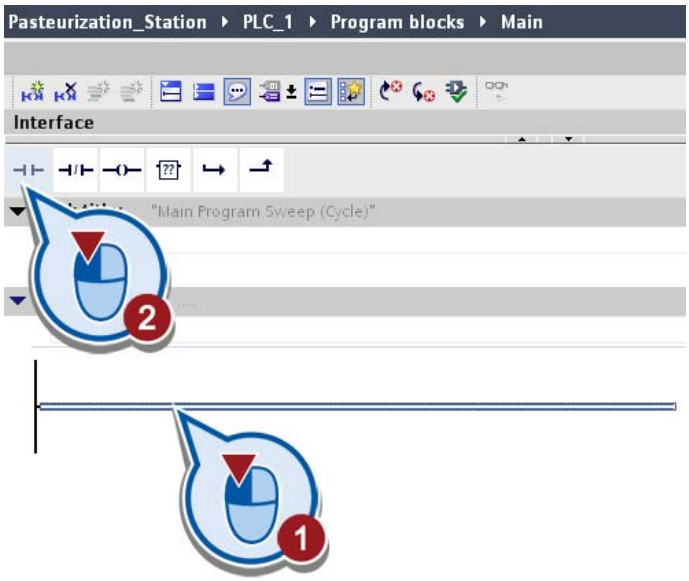
To create the program for switching the example machine on and off, follow these steps:

1. Activate the symbolic representation of the tag.

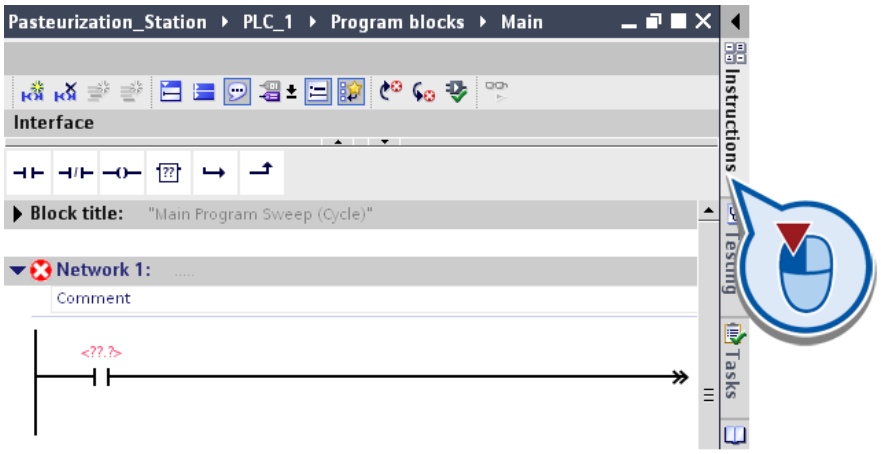


When the symbolic representation of the tags is activated, the addresses of the tags are not displayed in the network.

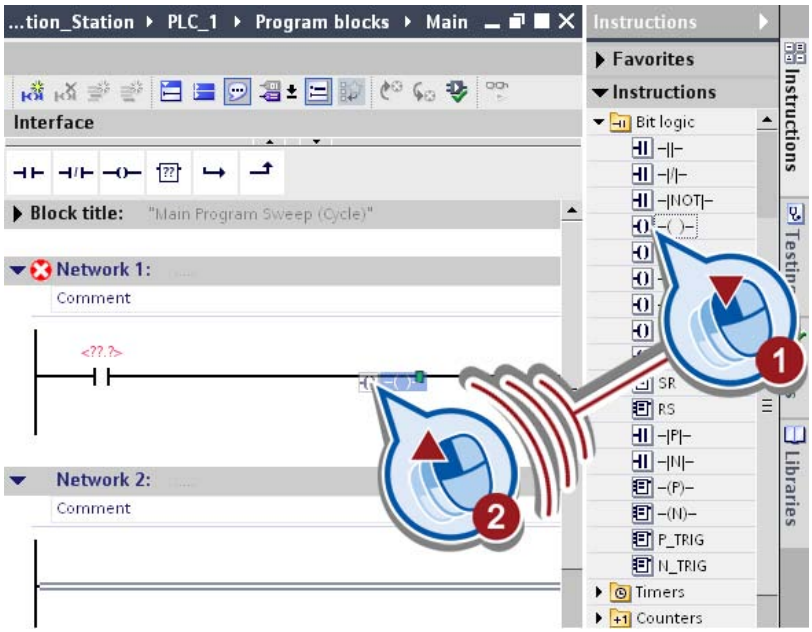
2. Insert a normally open contact into the first network of the block.



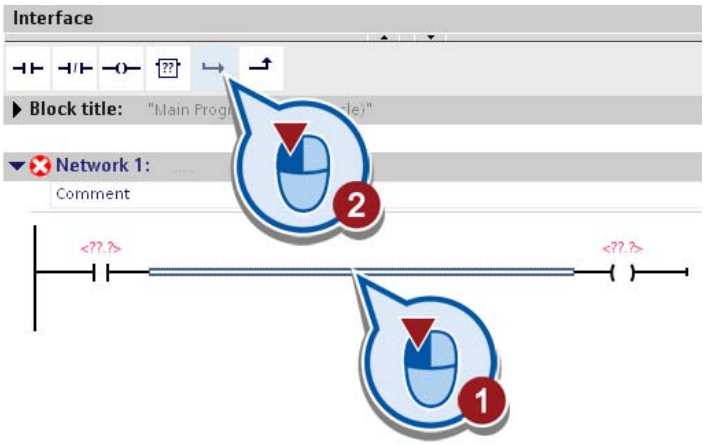
3. Open the "Instructions" task card.



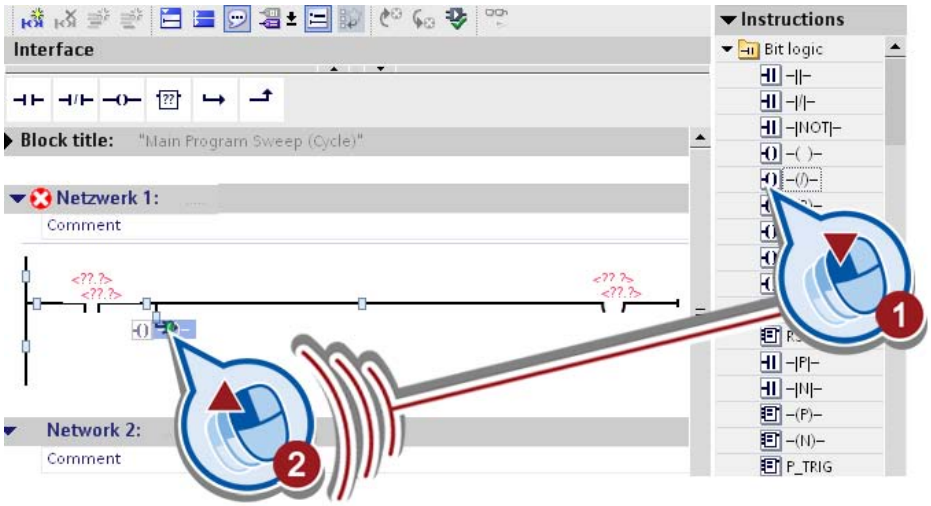
4. Insert the "Output coil" instruction at the end of the main rung.



5. Insert a branch.



6. Insert the "Negated coil" instruction at the end of the branch.



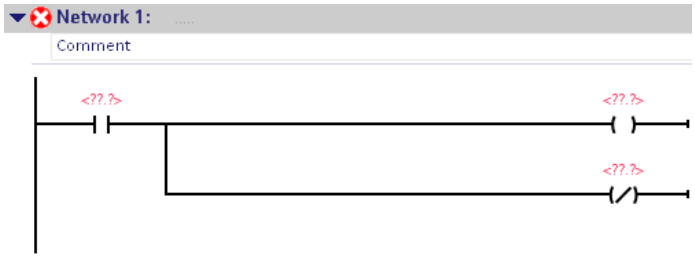
Note

Modifying an instruction

If you have selected the wrong instruction type, you can modify it directly in the network by means of the instruction icon. You open the drop-down list for selection of an instruction icon by moving the mouse pointer across the yellow triangle in the upper right-hand corner of the instruction.

Result

By inserting the LAD instructions in the first network of the organization block "Main [OB1]", you have created the first part of the program for switching the example machine on and off.



In the next section, you interconnect the LAD instructions with PLC tags and thereby close the program.

2.4.6 What are tags?

Introduction

In the previous step you have inserted three LAD instructions into the first network of the organization block "Main [OB1]". In the next section you will interconnect these instructions with PLC tags. The PLC tags are used to supply the instructions with values and the instructions are executed depending on these values.

In the following section you will learn more about the general function of tags in user programs.

Tags

A tag is a variable used in the program that can take on different values. Depending on the range of application, the tags are divided into the following categories:

- Local tags: Local tags apply only in the block in which they are defined.
- PLC tags: PLC tags apply throughout the entire PLC.

Most instructions in the program operate with tags. When you assign a tag to an instruction, the instruction is executed with the values of the specified tag.

The tags are managed centrally in the TIA Portal. It makes no difference if you create a PLC tag in the program editor or in the PLC tag table. If the tag is used at several points in the program or in the HMI screen, the changes to the tag are taken over immediately in all editors.

Benefits of tags

Tags have the advantage that they can centrally change an addressing used in the program. Without the symbolic addressing by tags, an addressing that is used again and again in the user program would have to be changed at several points in the program each time the configuration of the PLC inputs and outputs changes.

PLC tags

A PLC tag is made up of the following components:

- Name (e.g. CONVEYOR_ON): The name of a tag is valid for a PLC and may only occur once within the entire program and this particular PLC.
- Data type (e.g. BOOL): The data type defines the value representation and the permitted value range. By selecting the BOOL data type, for example, you specify that a tag can accept only the binary values "0" and "1".
- Address (e.g. M 3.1): The address of a tag is absolute and defines the memory area from which the tag reads or writes a value. Examples of possible memory areas are inputs, outputs and bit memories.

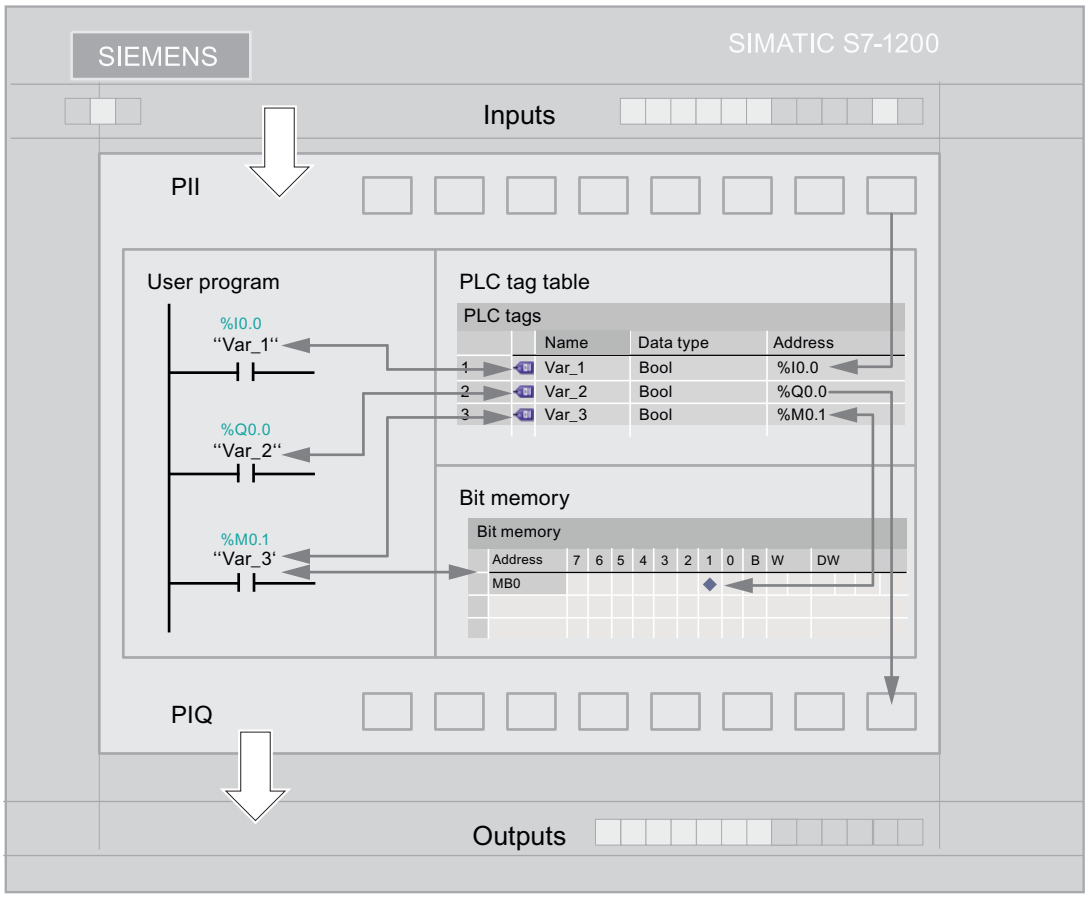
You can use the memory areas inputs (I) and outputs (Q) to address the process image.

The process image contains the image of the inputs and outputs of a PLC module:

- The signal states of the inputs in the process image are refreshed in each program cycle (see What are organization blocks? (Page 31)) by the operating system of the PLC.
- The signal states of the outputs in the process image are written to the outputs of the respective output module in each program cycle by the operating system of the PLC.

A machine or a process is generally controlled by means of the signal states of the outputs.

The following figure shows how PLC tag table, tags in the user program, bit memories, and inputs and outputs of the PLC are linked to each other in principle.



The bit memory area above all is used to save interim results. The values of the tags that are addressed in the bit memory area are stored in the system memory and not transferred to the modules. The data type of the tag determines how much memory a tag occupies in the memory. A tag of the BOOL data type, for example, occupies only one bit in the memory. A tag of the INT data type occupies 16 bits in the memory. Tags are not to overlap in a memory area. The address of a tag must be unique.

PLC tag table

The PLC tag table contains the definition of the tags and constants that are valid for a PLC. A PLC tag table is created automatically for each PLC created in the project.

The following table shows the meaning of the individual table columns of the "Tags" tab:

Column	Description
	Symbol you can click in order to move a tag into a network via a drag-and-drop operation for use as an operand.
Name	Unique name within the PLC that you assign to the tag.
Data type	Data type that you specify for the tag.
Address	Tag address.
Retentivity	The values of retentive tags are retained even after the power supply is switched off.
Monitor value	Current data value in the PLC. This column only appears if an online connection is available and you select the "Monitoring" button.
Comment	Comments to document the tags.

2.4.7 Defining and interconnecting PLC tags

Introduction

In the TIA Portal you have the options of directly creating tags when you create the user program in the networks. The following steps show you how to define PLC tags and interconnect the inserted LAD instructions with PLC tags. Depending on the tag values, the LAD instructions are executed and thereby control the switching on and off of the machine.

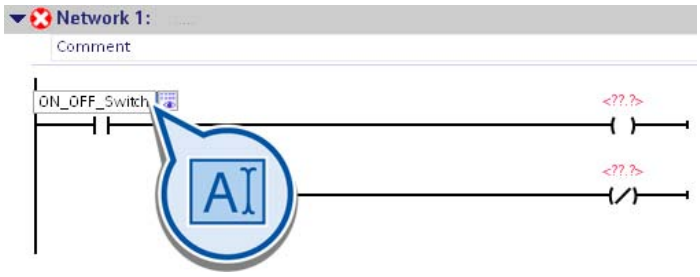
Requirements

- The project has been created.
- The PLC is configured.
- The block "Main [OB1]" is open.
- The LAD instructions "Normally open contact", "Output coil" and "Negated coil" are inserted in the first network of the organization block "Main [OB1]".

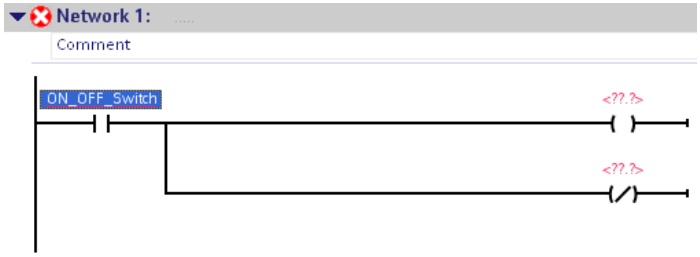
Procedure

To interconnect the LAD instructions with PLC tags, follow these steps:

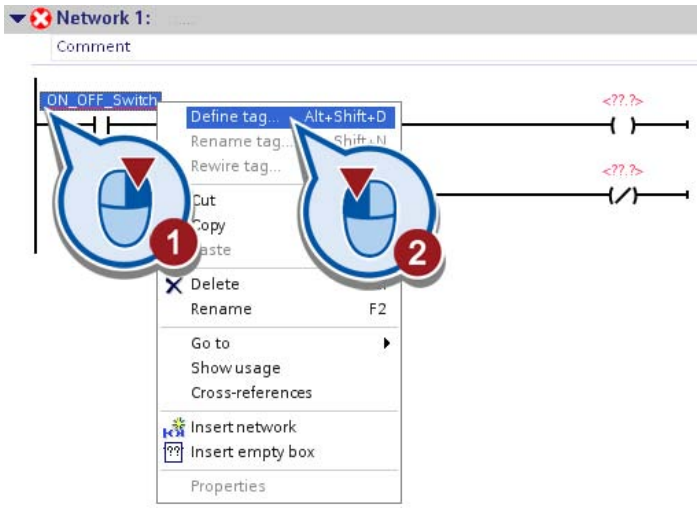
1. Open the first network of the organization block "Main [OB1]".
2. Enter the name "ON_OFF_Switch" in the operand placeholder of the normally open contact.



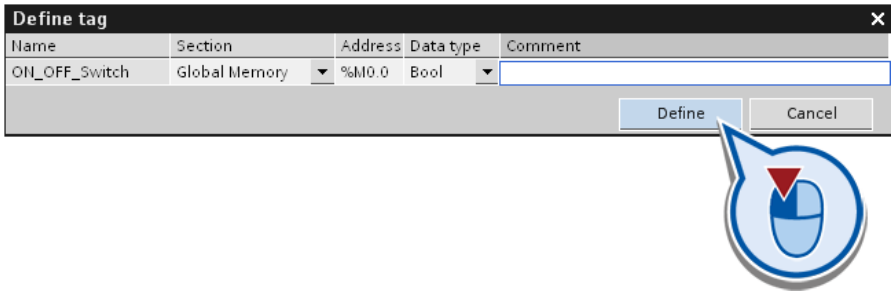
3. Confirm the entry with the Enter key.



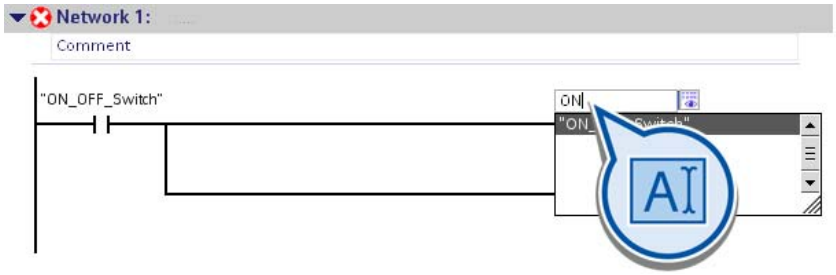
4. Open the "Define tag" dialog box.



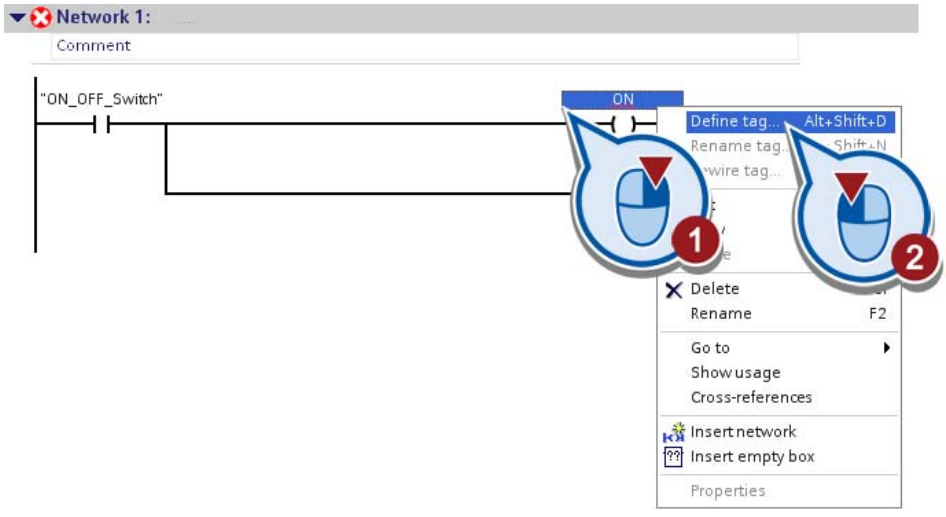
5. Define the "ON_OFF_Switch" tag.



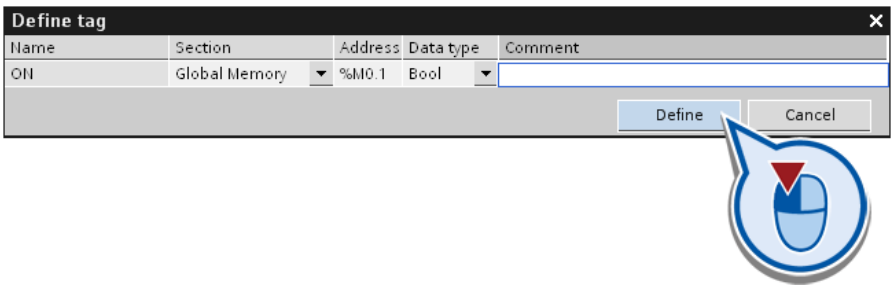
6. Enter the name "ON" in the operand placeholder of the "Output coil" instruction.



7. Confirm the entry with the Enter key.
8. Open the "Define tag" dialog box.



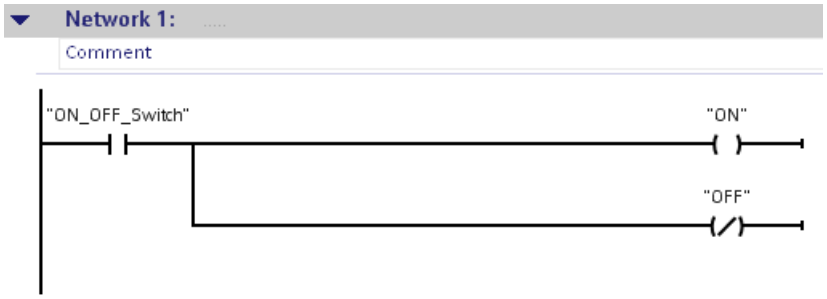
9. Define the "ON" tag.



10. Enter the name "OFF" in the operand placeholder of the "Negated coil" instruction and define the corresponding tag.
11. Click the "Save project" button on the toolbar to save the project.

Result

You have programmed a pushbutton switch for switching the example machine on and off.



The actuation of the pushbutton switch has the following effects:

- When the pushbutton switch is pressed once, the "ON_OFF_Switch" tag is set to the signal state "1".
  - The power is relayed and the "ON" tag is set to the signal state "1" by the "Output coil" instruction.
  - The machine is switched on.
  - The "OFF" tag supplies the signal state "0" and has no further effect.
- When the pushbutton switch is pressed a second time, the "ON_OFF_Switch" tag is set to the signal state "0".
  - The current flow is interrupted and the "OFF" tag is set to the signal state "1" by the "Negated coil" instruction.
  - The machine is switched off.
  - The "ON" tag supplies the signal state "0" and has no further effect.

2.5 Test program

2.5.1 Loading the program to the target system

Introduction

The following steps show you how to load the program to the PLC. An online connection is established during the loading process between your programming device (PG) or programming computer (PC) and the PLC. The program that is stored on the hard disk of your programming device (PG) or programming computer (PC) is written to the memory of the PLC during the loading step. The blocks contained in the program are compiled during the loading process so that they can be processed by the PLC. Once the program is compiled and loaded, it can be processed by the PLC.

Note

Online/offline comparison

Changes made to the program on the programming device / PC after the loading process are not registered by the PLC. You can use the TIA Portal for an online/offline comparison of your project data and to display any deviations. In online mode you can determine with the help of the icons in the project tree if the "Offline" program components on your programming device / PC are identical with the "Online" programming elements on the PLC. To refresh the status of the program on the PLC, you have to re-load the program.

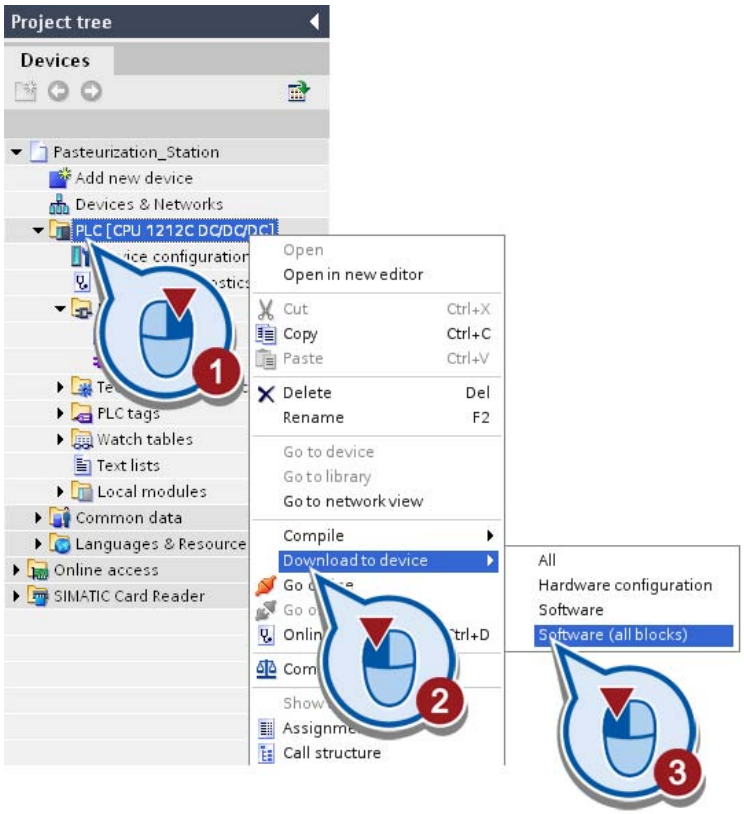
Requirements

A connection is established between the programming device / PC and the PLC (see "Additional Information").

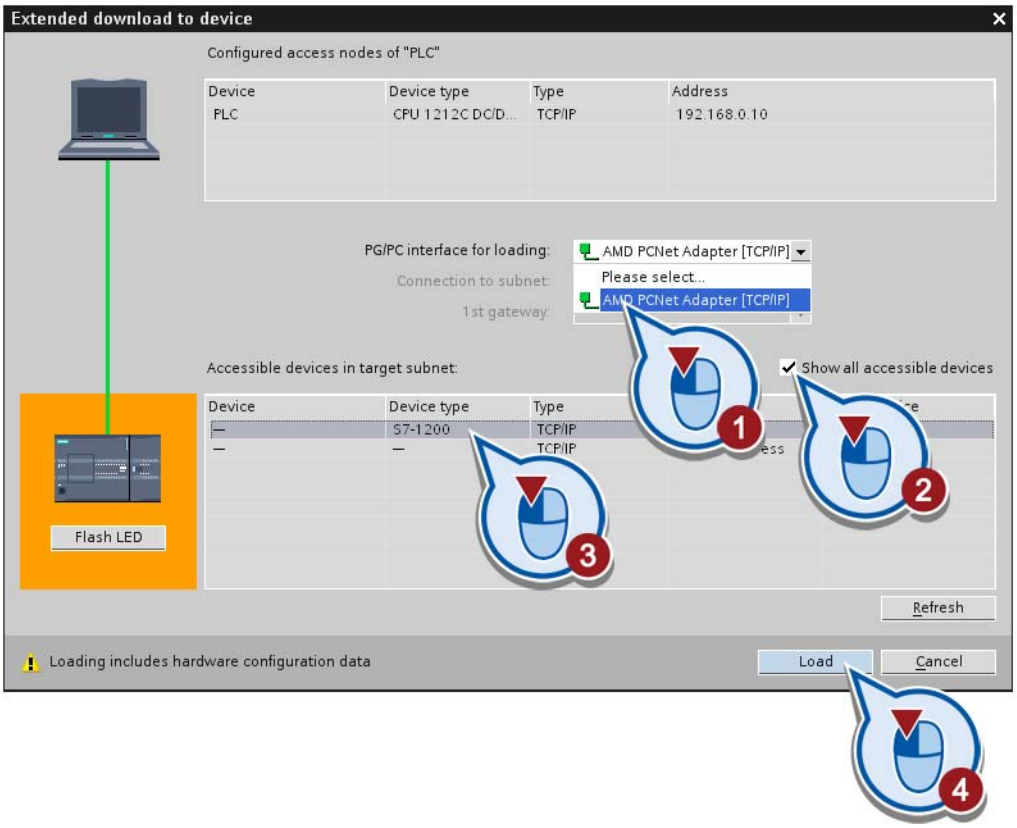
Procedure

To load the program to the PLC, follow these steps:

1. Start the load process.



2. Select the interface you want to use to connect the device. Activate the check box "Show all accessible devices".
   All devices that can be accessed via the selected interface are displayed under "Accessible devices in target subnet". Select the PLC and load the user program.



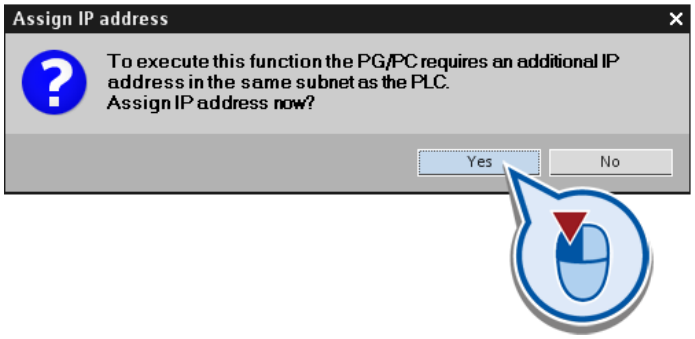
Note

Check online connection

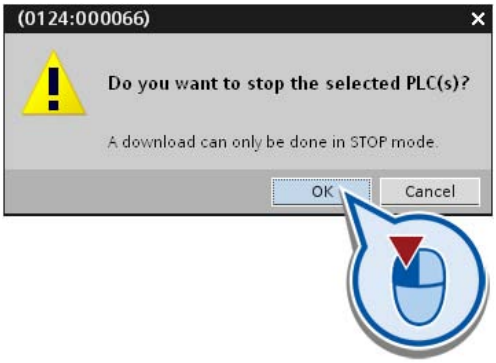
If no device is displayed in the "Accessible devices in target subnet" list, it can have several reasons:

- There is a problem with the hardware connection of the PLC.
- There is a problem with the Ethernet interface on your programming device / PC.
- The IP address of the PLC must be in the same subnet as the IP address of the programming device / PC.

3. Acknowledge assignment of the correct IP address if it has not been assigned yet.

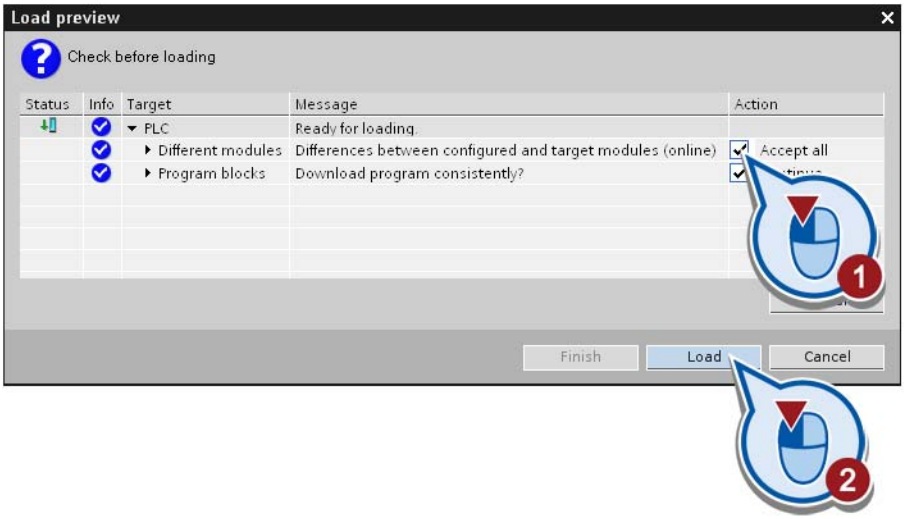

Assign IP address
To execute this function the PG/PC requires an additional IP address in the same subnet as the PLC.
Assign IP address now?
Yes
No

4. If the PLC is in "RUN" mode, set the PLC to "STOP" mode in the following window:



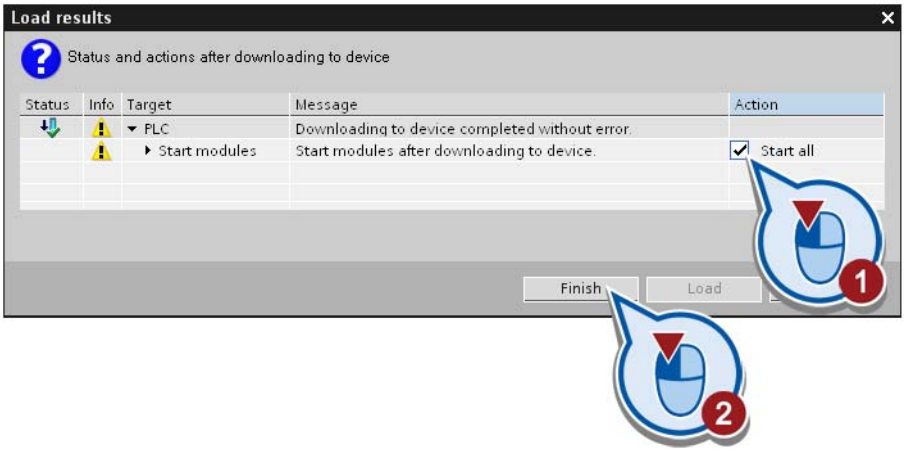
The "Download preview" dialog opens.

5. If there are differences between the configured modules and the target modules, activate the corresponding check box to accept the differences. Click the "Load" button. Make sure that the "Continue" check box is selected.

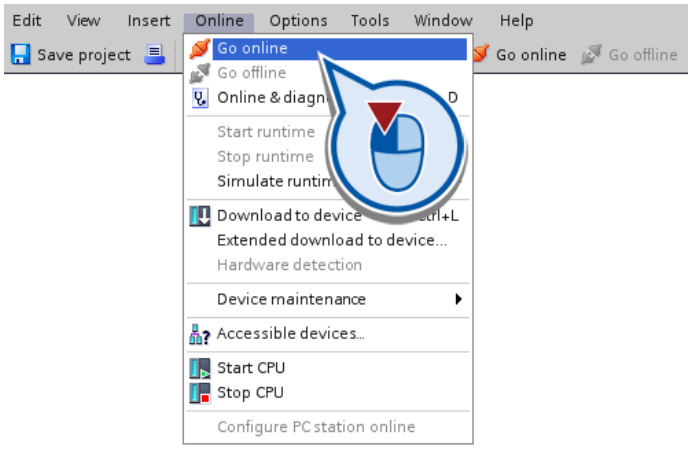


The program is loaded and the actions during the load process are displayed. The "Download results" dialog box opens after the download process has been completed.

6. Start the module.



7. Activate the online connection.



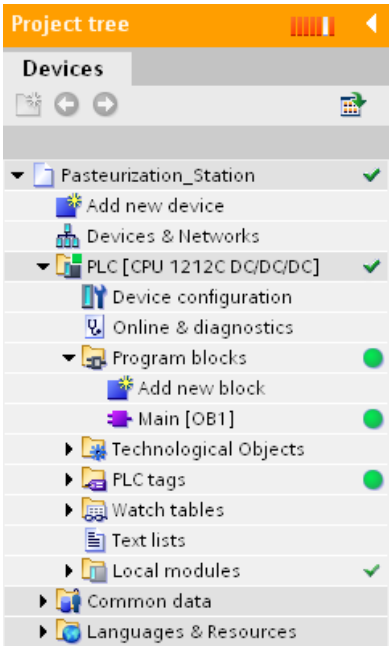
Note

Correcting compilation errors

If errors in the program are detected during the compilation, you can have these errors and notes displayed in the Inspector window under "Info > Compile" to correct them.

Result

You have loaded the program to the PLC. The status of the program components is shown in the project tree. The green icons indicate that the "offline" and "online" program elements are identical. You will find the meaning of the additional status icons in the respective tooltip.



Note

Downloading blocks

Loading the blocks from the project tree is not the only way to transfer the blocks to the PLC. You can use drag-and-drop to store the blocks in the list of accessible devices in the project tree. If you use the button "Load to device" in the toolbar, you will load those blocks that are open in the editor or that you have selected in the project tree.

Additional information

Information on establishing and configuring the Ethernet connection is available in the following documentation:

- "SIMATIC S7-1200 Automation System" system manual
- TIA Portal information system, "Setting parameters for the Ethernet interface" chapter
- TIA Portal information system, "Configuring Industrial Ethernet" chapter

Information on establishing the Ethernet interface on the programming device / PC is available in the documentation of the operating system you are using and in the documentation of the network card.

2.5.2 Test program with program status

Introduction

The following steps show you how to test the created program with program status. If you display the program status, you can monitor the execution of the program. You can enable the status starting at a specific location in the program and you then obtain an overview of the values of the individual tags and the results of logic operations. This allows you to check whether or not the components of the automation system are being correctly controlled.

The display of the program status is updated cyclically. It begins at the selected network.

In the program status you can allocate values to the tags by performing one of the following actions using the "Modify" command of the shortcut menu:

- Modify to 1: Use this command to set the tags of the BOOL data type to the signal state "1".
- Modify to 0: With this command you set the tags of the BOOL data type to the signal state "0".
- Modify value: You can enter a modify value for tags that are not of the BOOL data type.

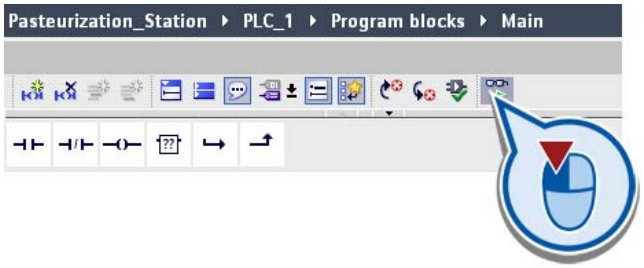
Requirements

- The PLC is configured.
- There is no voltage at the inputs and outputs of the PLC, because the modified values are overwritten by the module in online mode.
- The organization block "Main [OB1]" is opened in the program editor.

Procedure

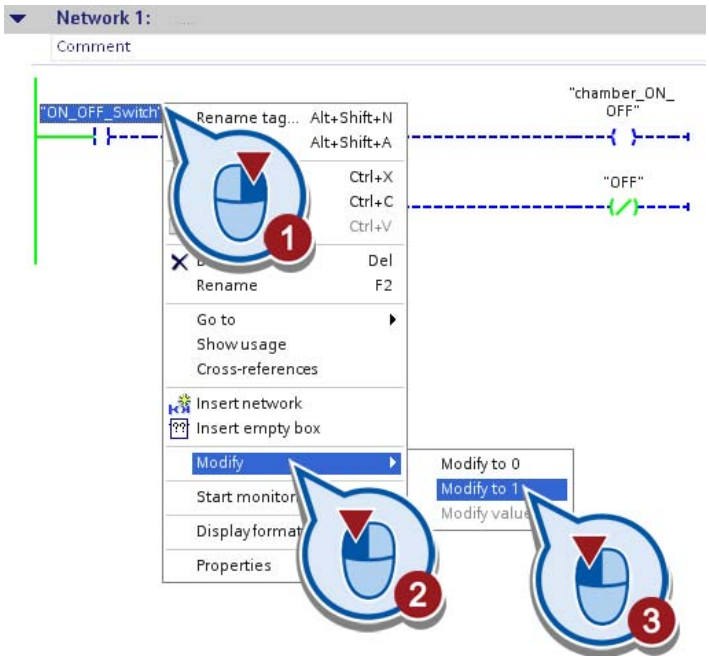
To test the created program with program status, follow these steps:

1. Enable monitoring.

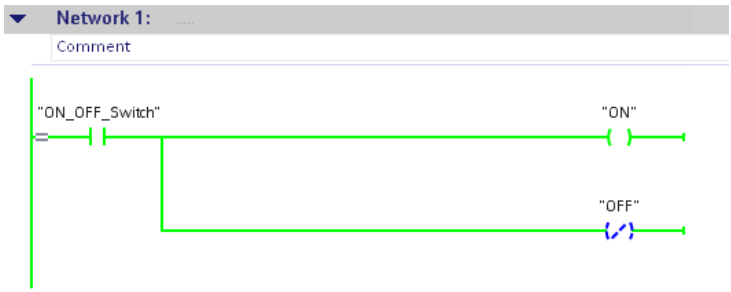


The program status is displayed.

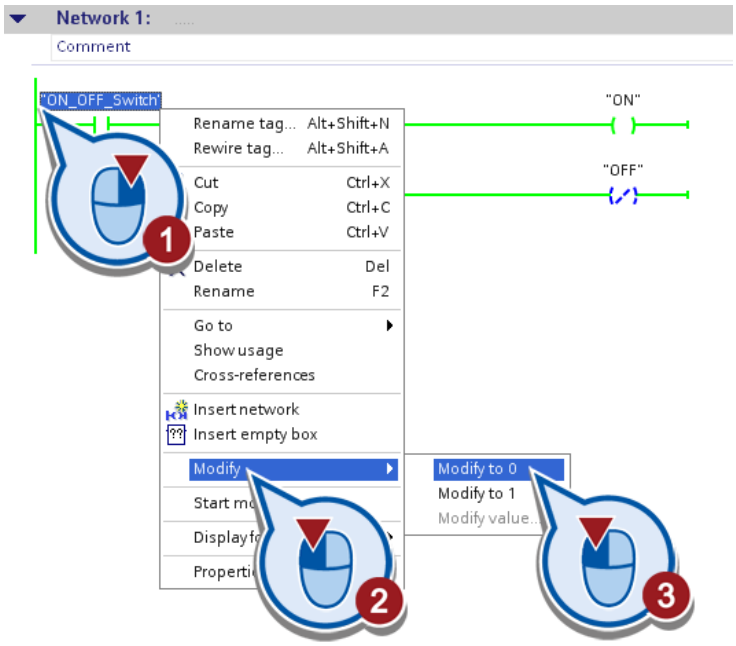
2. Modify the "ON_OFF_Switch" tag to "1".



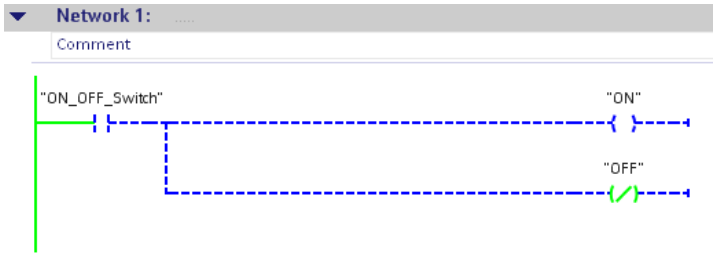
If the "ON_OFF_Switch" tag is set to the signal state "1", the normally open contact is closed. The current flows through the normally open contact to the coils at the end of the network. The current flow is indicated by the green color of the current path. The "ON" tag is set and the example machine will be switched on. The "OFF" tag has the signal state "0" and no further effect. This status is indicated by the blue dashed line.



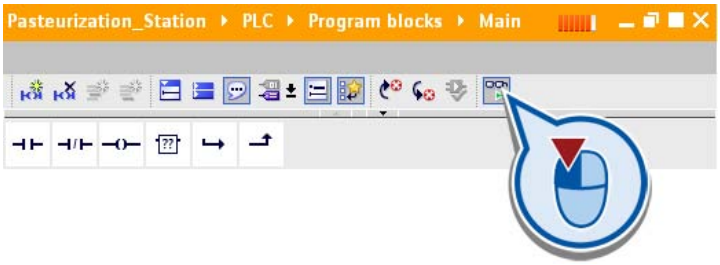
3. Modify the "ON_OFF_Switch" tag to "0".



The "ON_OFF_Switch" tag is reset to the signal state "0". The current flow to the coils at the end of the network is interrupted. The "OFF" tag is set. The "ON" tag is reset to "0".



4. Disable the program status.



5. Separate the online connection.

Result

You have tested the program and checked the correct execution of the program.

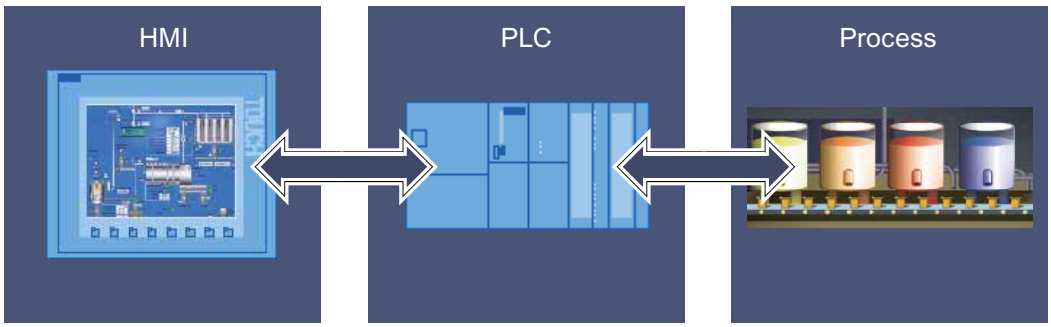
In the next section you will program a HMI screen to visualize the program steps.

2.6 Create HMI screen

2.6.1 Visualization in the TIA portal

HMI - Human Machine Interface

An HMI system represents the interface between the user and the process. The process operation is controlled mainly by the PLC. The user can use the HMI device to monitor the process or intervene in the running process.



You have the following options for operating and monitoring machines and plants:

- Display processes
- Operate processes
- Output alarms
- Administer process parameters and recipes

2.6.2 Create HMI device with HMI screen

Introduction

The following steps show you how to create a new HMI device and a template for the HMI screen.

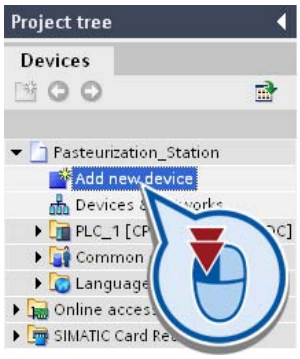
Requirements

- The program has been created.
- The project view is open.

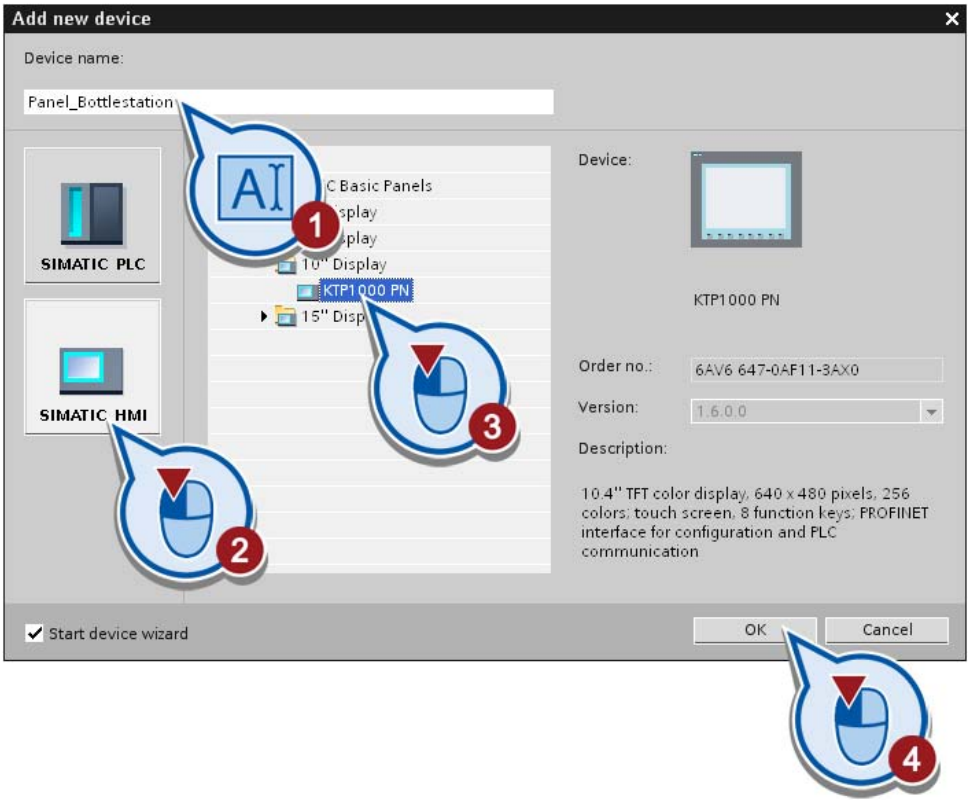
Adding a new HMI device

To add an HMI device, follow these steps:

1. Use the project tree to add a new device.



2. Assign a name and select an HMI device. Leave the "Start device wizard" check box selected.

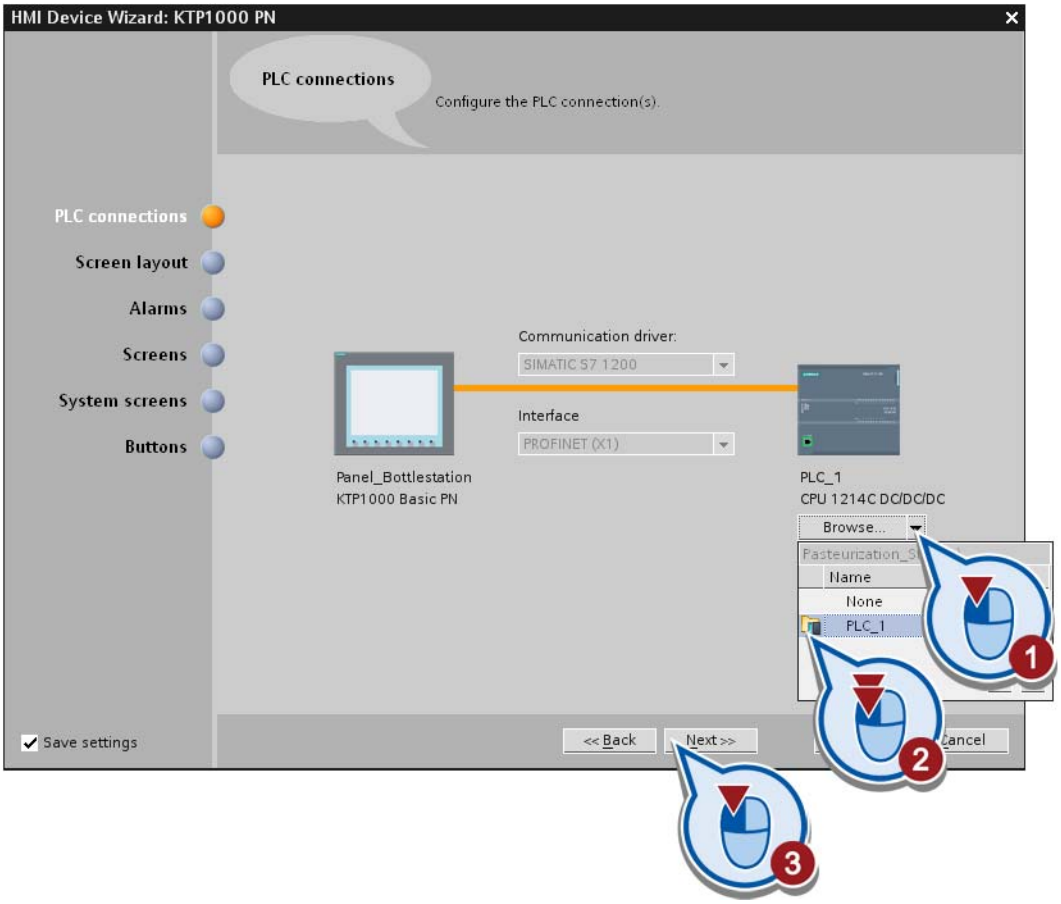


Creating a template for an HMI screen

The HMI device wizard opens after you have created an HMI device. The HMI device wizard starts with the "PLC connections" dialog box.

To create a template for the HMI screen, follow these steps:

1. Configure the connection to the PLC.

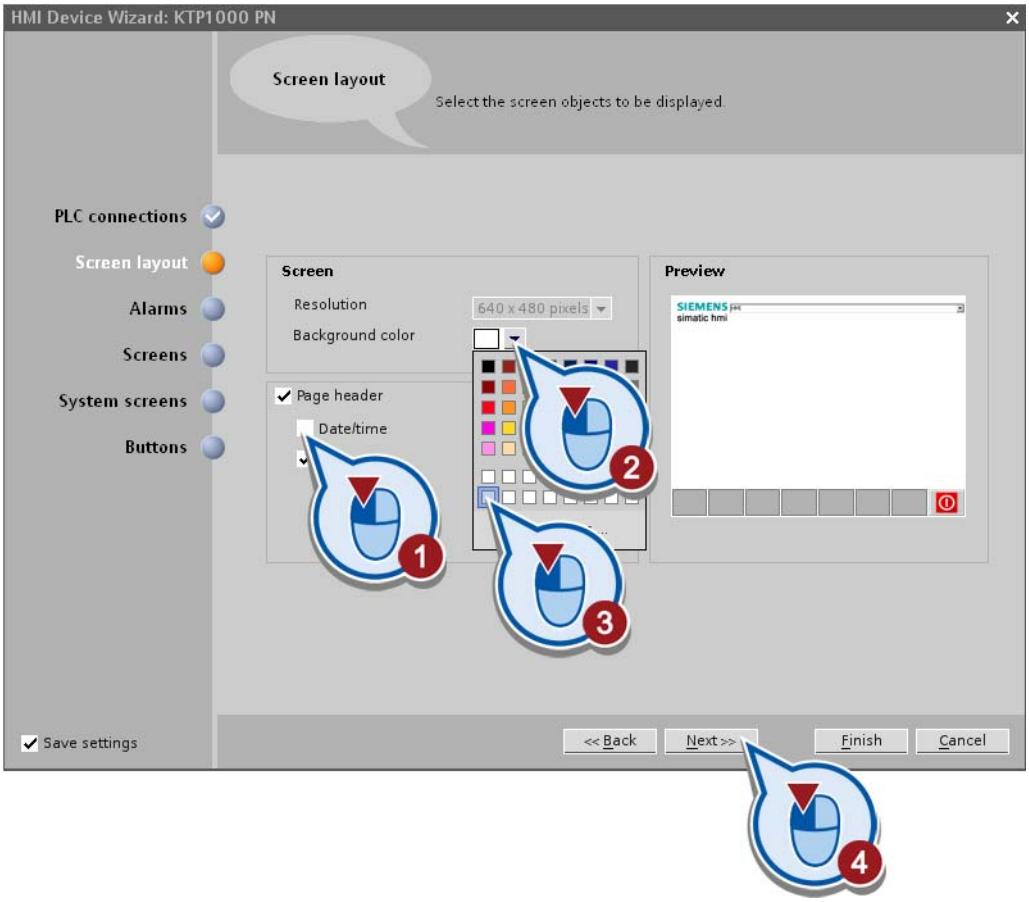


Note

Subsequently configuring the PLC connection

You can also configure the connection between the HMI device and the PLC under Devices & Networks. If you configure the connection in this dialog box, it will be created automatically.

2. Select the background color for the template and the elements for the header.

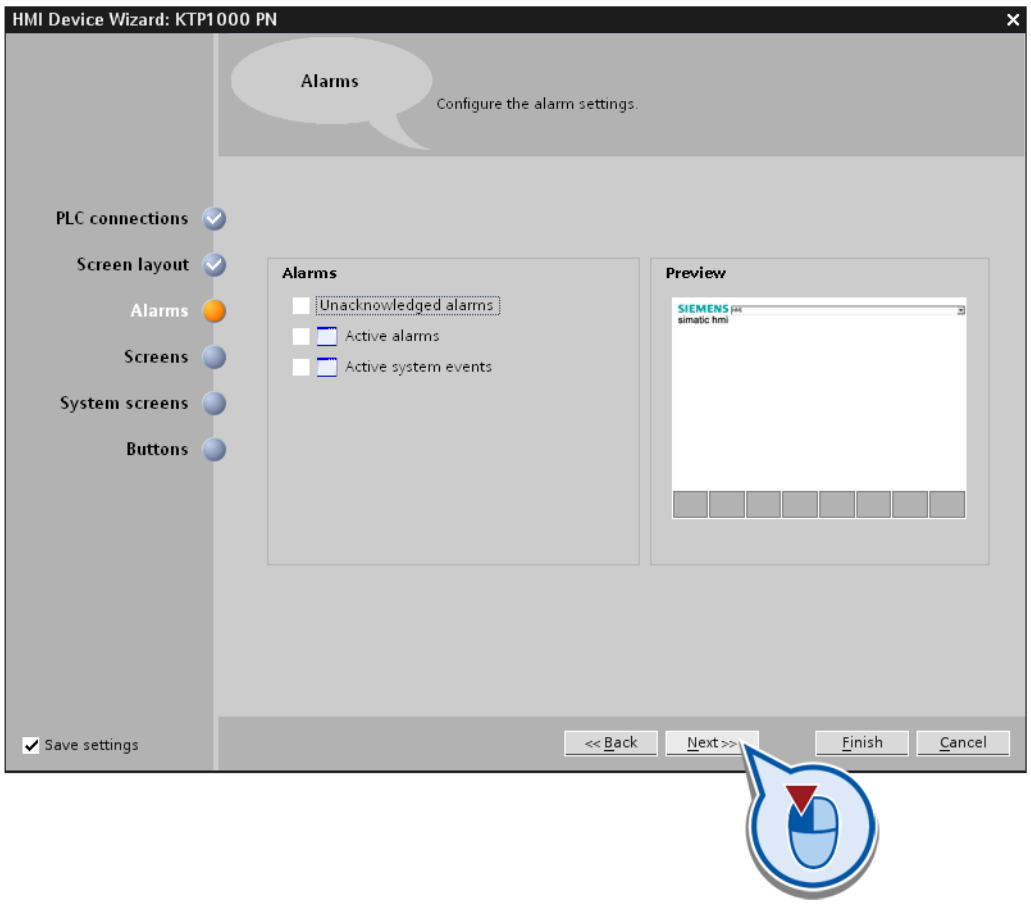


Note

Subsequently changing the display

In the template of the HMI screen you can subsequently change the display settings made here.

3. Disable the alarms. These are not necessary for the example project.



Note

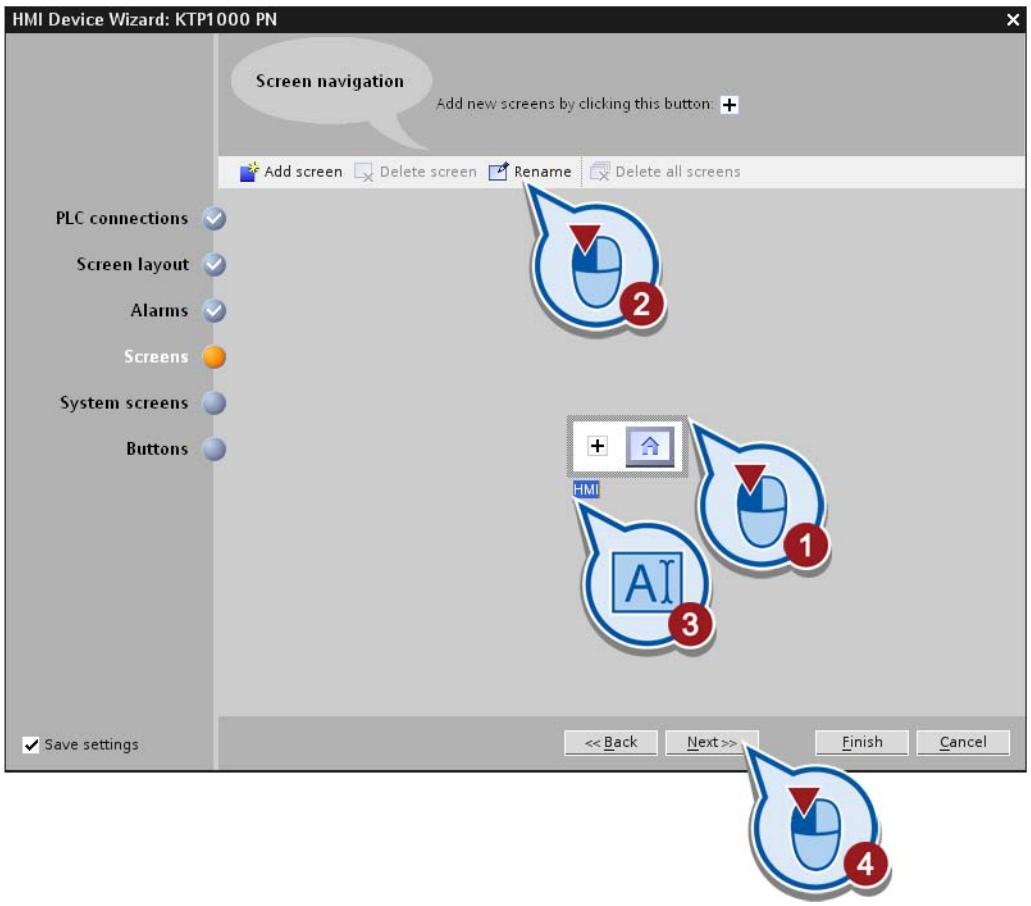
Alarms

If you enable the alarms via the HMI device wizard, you will be able to output the alarms via the HMI device. The alarm windows created here will be created in the global screen under "Screen management". Alarms are used, for example, to output warnings via the HMI device when a limit is exceeded. Alarms can be supplemented by any additional information, for example, to make it easier to localize faults in the system. A basic distinction is made between user-defined alarms and system alarms:

- User-defined alarms are used to monitor the machine process.
- System alarms are imported into the project and contain status information of the HMI device used.

Additional information on alarms is available in the information system of the TIA Portal.

4. Rename the screen in which the graphic elements will later be created to "HMI".

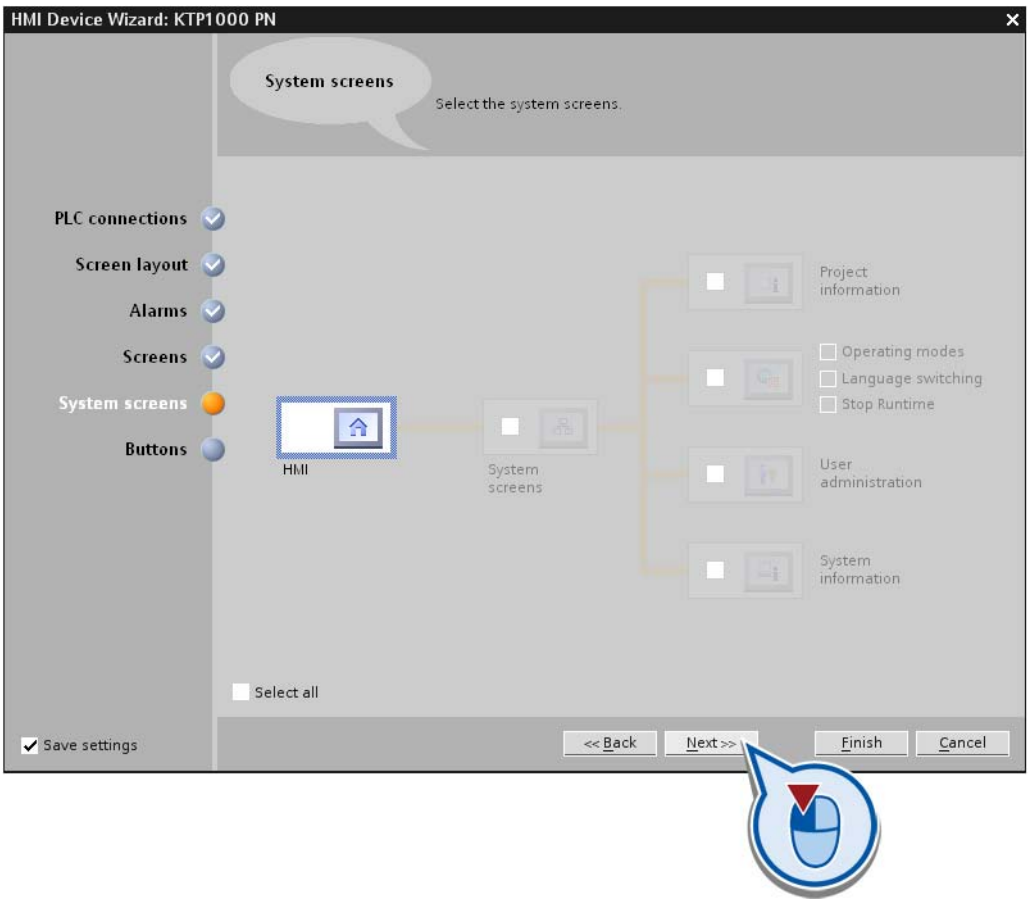


Note

Screen navigation

You can use this dialog box to create screens in more extensive projects and to build up a screen navigation. Buttons are automatically created for navigating between the screens.

5. Disable the system screens. These are not necessary for the example project.

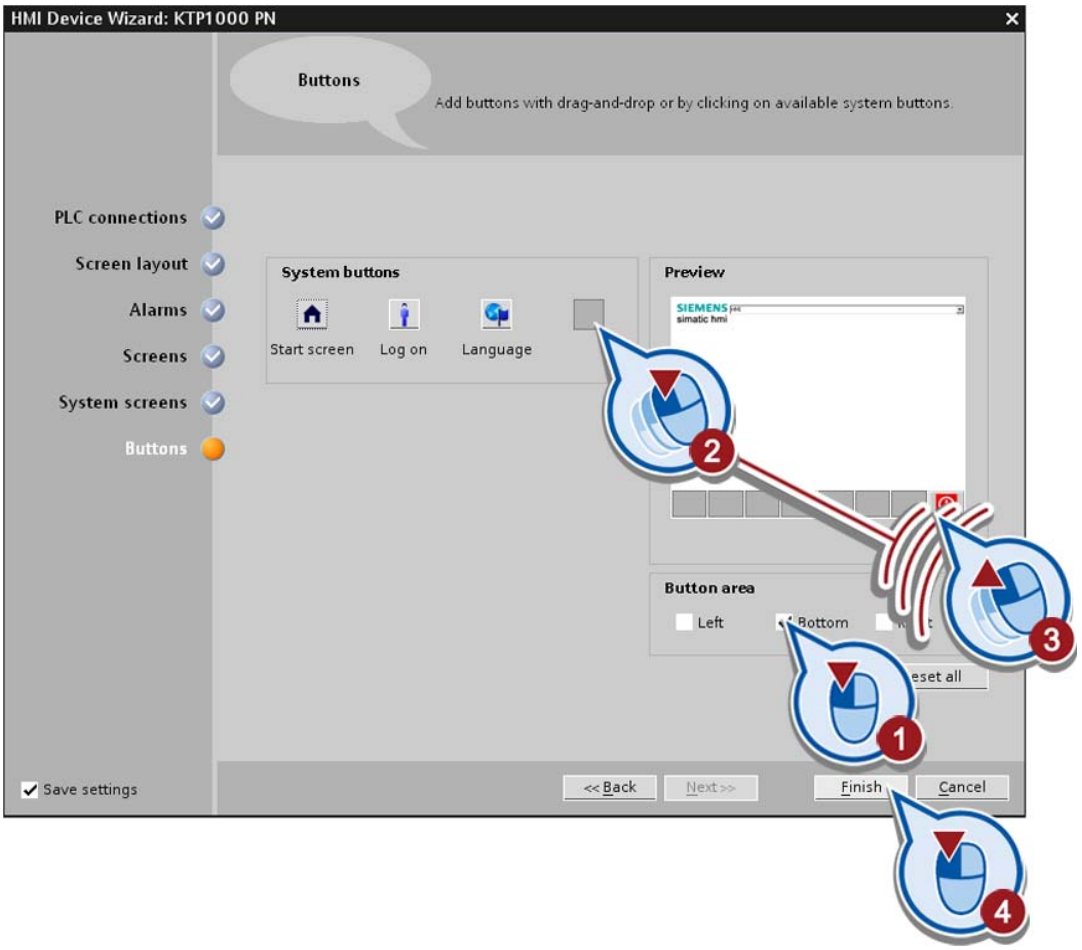


Note

System screens

You can use the system screens to set up project, system and operating information as well as the user administration as HMI screens. The buttons for navigating between the main screen and the system screens are created automatically, as with the screen navigation.

6. Enable the lower button area and insert the "Exit" button. Runtime can be terminated using this button.



7. Click the "Save project" button on the toolbar to save the project.

Result

You have created an HMI device in the project and a template for the HMI screen. In project view the created HMI screen will be displayed in the editor.

In the next section you create screen objects in the HMI screen to visualize the programmed processes.

2.6.3 What are graphic objects?

Introduction

You can use the TIA portal to create screens for operating and monitoring machines and plants. Predefined objects are available to help you create these screens; you can use these objects to simulate your machine, display processes and define process values. The functions of the HMI device determine project visualization in HMI and the functional scope of the graphic objects.

Graphic objects

Graphic objects are all those elements that can be used for the visualization of the project in HMI. These include, for example, texts, buttons, diagrams or graphics for visualizing machine parts.

Use of graphic objects

Graphic objects can be statically visualized or used as dynamic objects with the help of tags:

- Static objects do not change in runtime. In the Getting Started project a conveyor is created as static object.
- Dynamic objects change depending on the process. You visualize current process values by means of:
  - PLC tags from the memory of the PLC
  - Internal tags from the memory of the HMI device in the form of alphanumeric displays, trends and bars.

  Dynamic objects include text boxes on the HMI device to exchange process values and operator inputs between the PLC and HMI device by means of tags.

2.6.4 Creating and configuring graphic objects

2.6.4.1 "Machine ON/OFF" button

Introduction

The following steps show you how to create the "Machine ON/OFF" button and how to connect it to the PLC tag "ON_OFF_Switch" using an external HMI tag. You can use this approach to modify the process values of the PLC tag by means of the HMI screen.

External HMI tags

You use an external HMI tag to access a PLC address. This allows you, for example, to input a process value via an HMI device or to directly modify process values of the control program via a button. Addressing takes place via the PLC tag table in the PLC linked to the HMI device. The PLC tag is linked to the HMI tag with the symbolic name. This means you do not have to adapt the HMI device when you change the address in the PLC tag table.

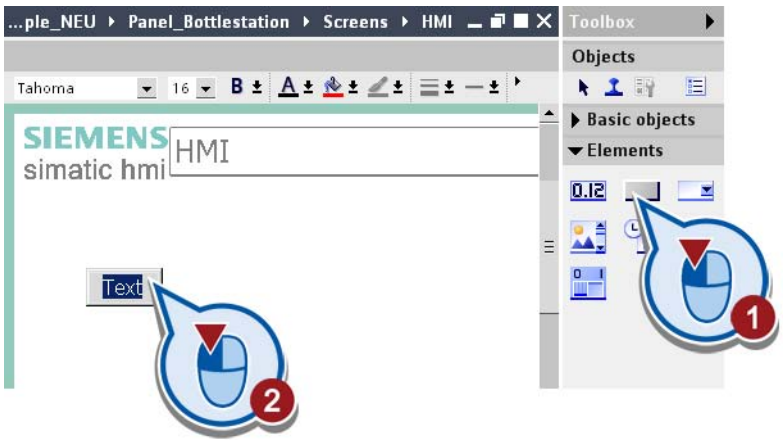
Requirements

The HMI screen is open.

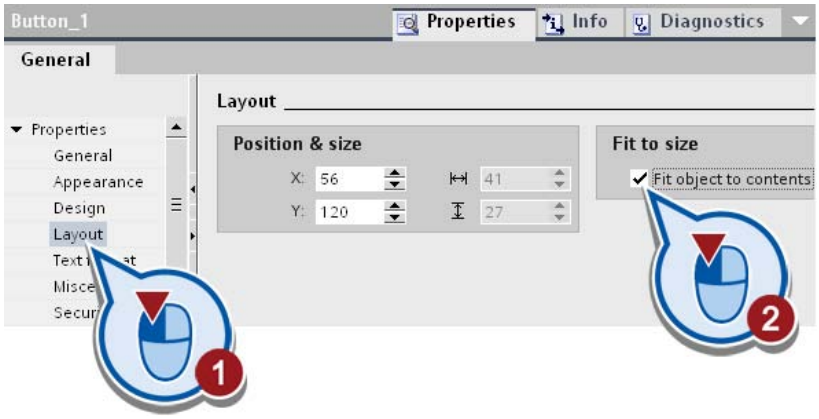
Procedure

To connect the "Machine ON/OFF" button with the PLC tag "ON_OFF_Switch", follow these steps:

1. Delete the standard text field "Welcome..." from the HMI screen.
2. Create a button.



3. In the Inspector window, select the "Fit object to contents" option, to automatically adjust the button size to the text length.

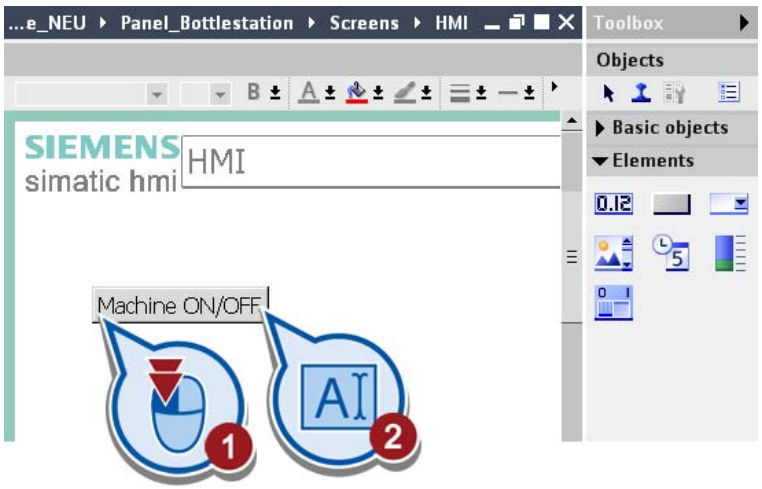


Note

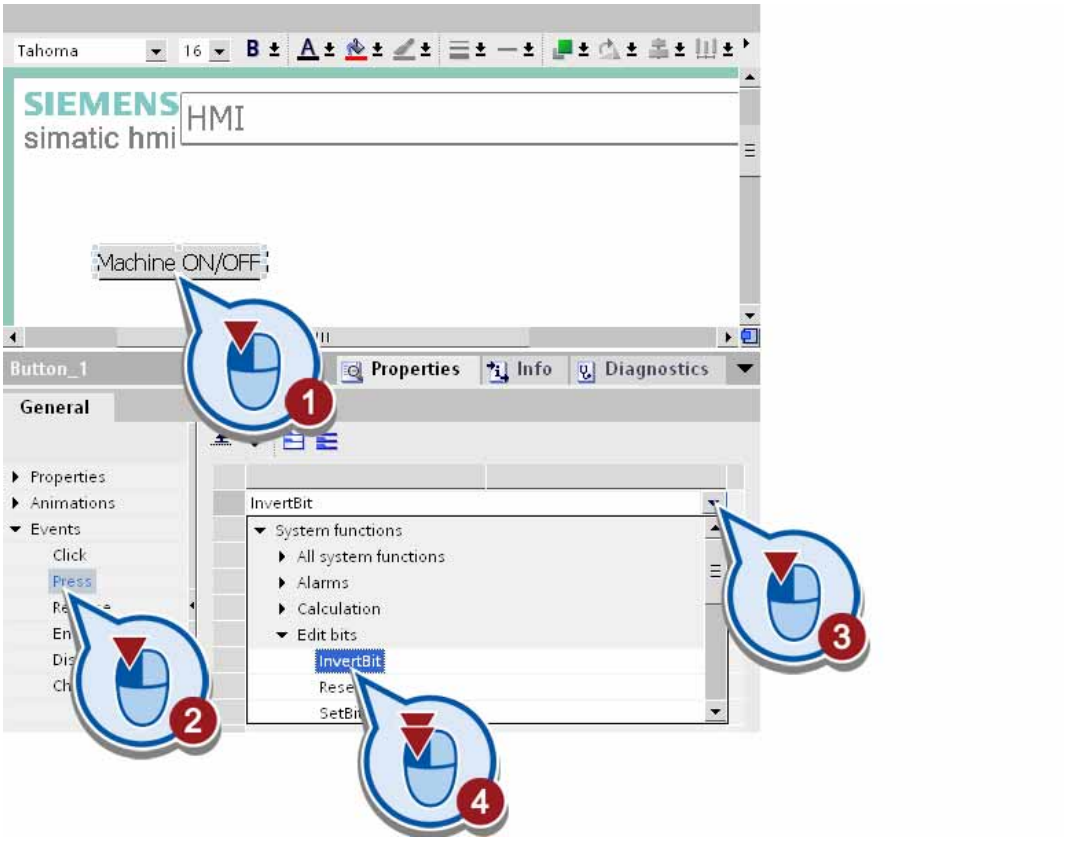
"Fit object to contents" function

You can use this function in particular when you are working in future projects with language selection for the HMI screens. Depending on the selected language, the translated text can be shorter or longer than the original. Use this function to ensure that the button labels are not truncated. The button size will adjust automatically in case of text changes in the original.

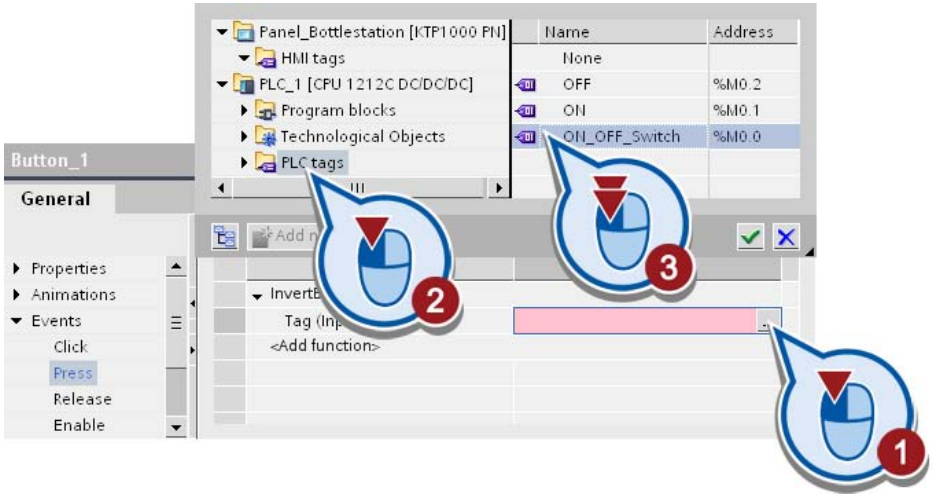
4. Label the button with the text "Machine ON/OFF".



5. Assign the event "Pressing", which triggers the function "InvertBit", to the button.



6. Link the "InvertBit" function to the PLC tag "ON_OFF_Switch".



Note

HMI connections are automatically created in the TIA Portal

If you did not configure a connection between the HMI device and the PLC, the connection will be automatically created as soon as you link a PLC tag to an HMI object.

Result

You have connected the "Machine ON/OFF" button with the PLC tag "ON_OFF_Switch". When you press the button on the HMI device, the bit of the PLC tag is set to the value "1" (machine on). When you press the button a second time, the bit of the PLC tag is set to the value "0" (machine off).

2.6.4.2 Graphic objects "LEDs"

Introduction

The following steps show you how to set up two status LEDs (red/green) using the circle object and animate them depending on the value of the PLC tag "ON_OFF_Switch".

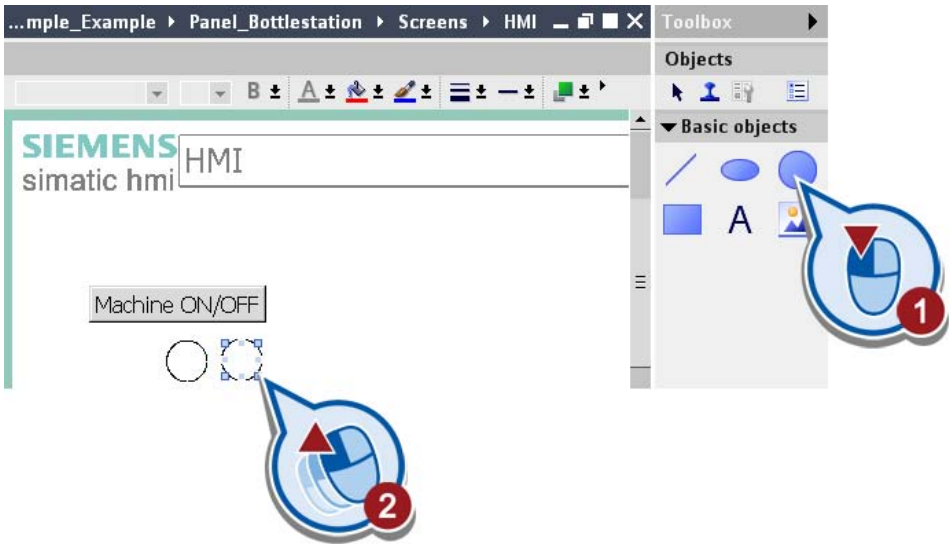
Requirements

The HMI screen is open.

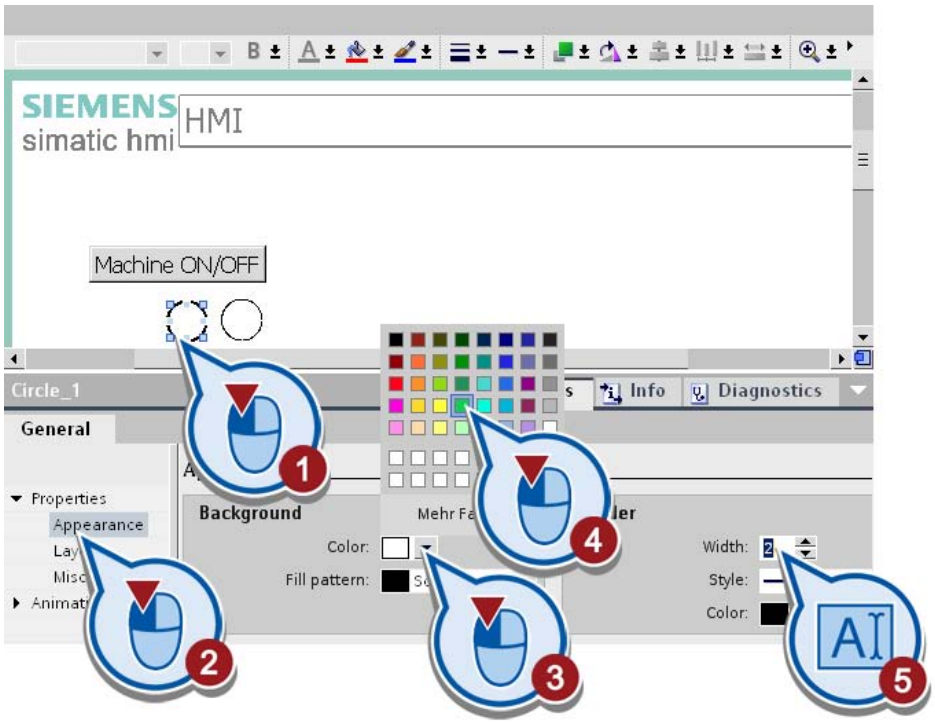
Procedure

To create and animate the LEDs, follow these steps:

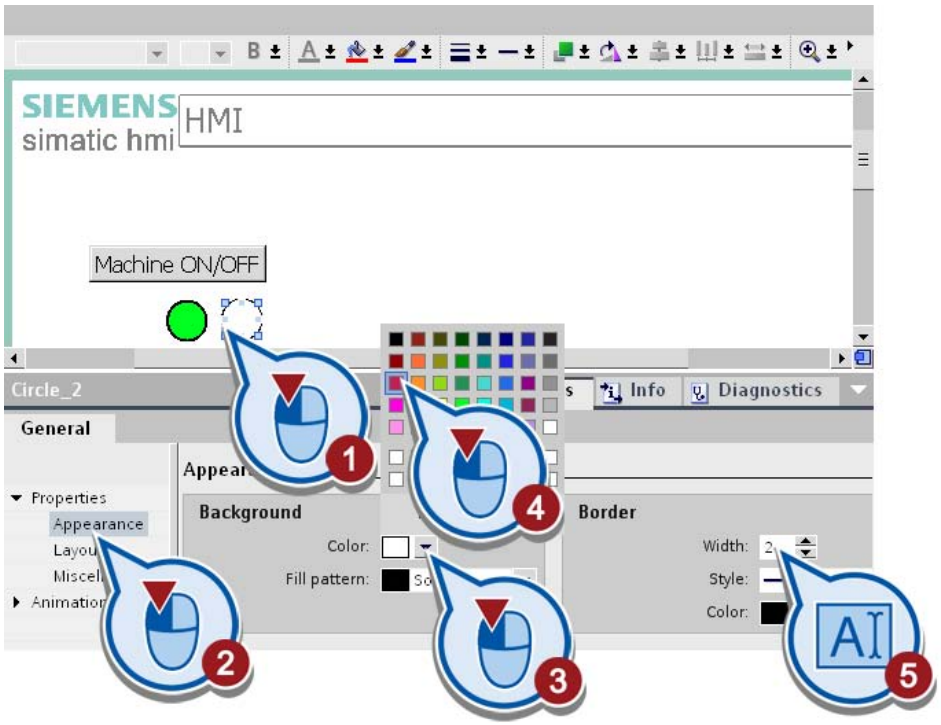
1. Keeping the Shift key pressed, draw two circles beneath the "Machine ON/OFF" button.



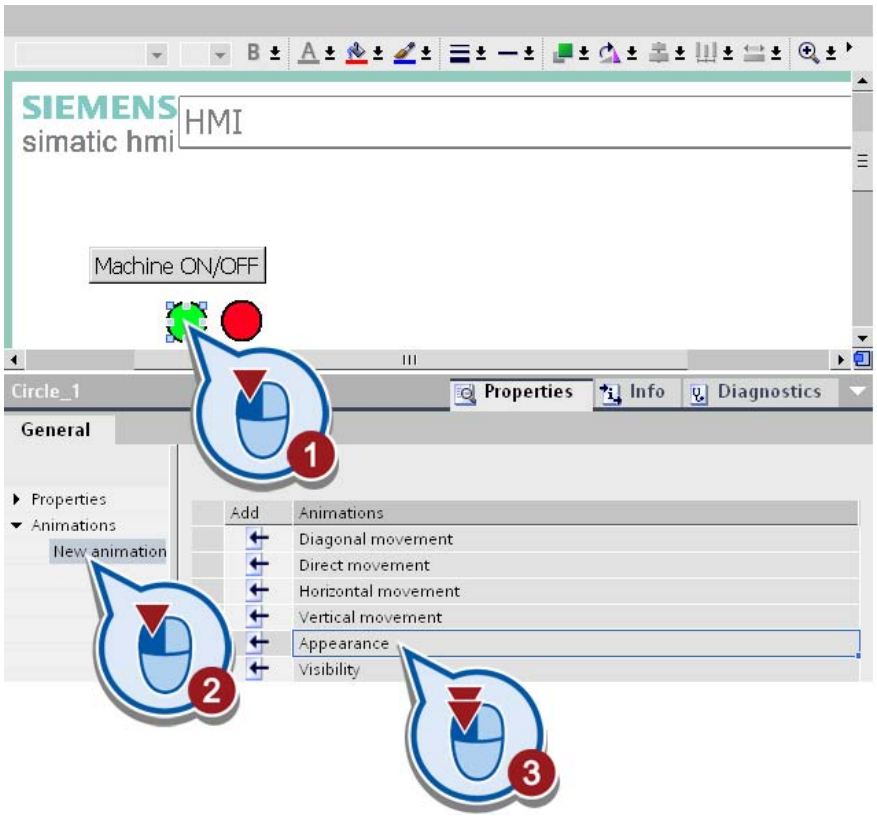
2. Assign the background color green and a border of "2" to the first circle.



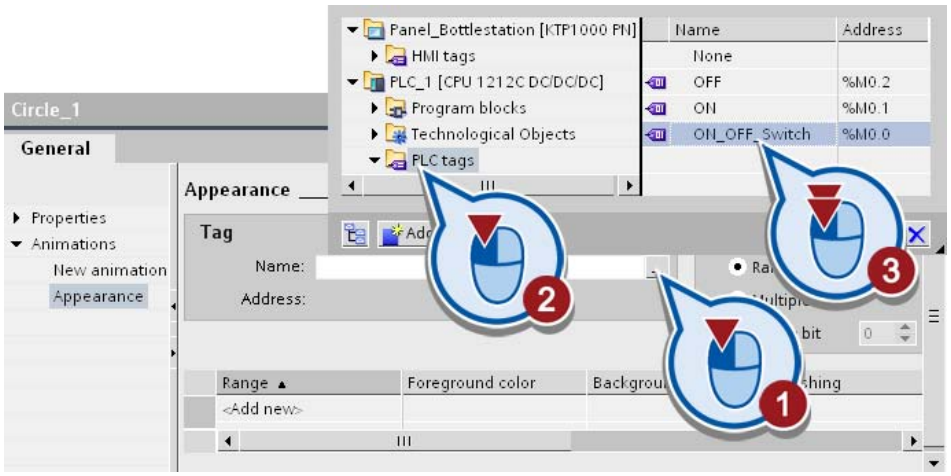
3. Assign the background color red and the same border of "2" to the other circle.



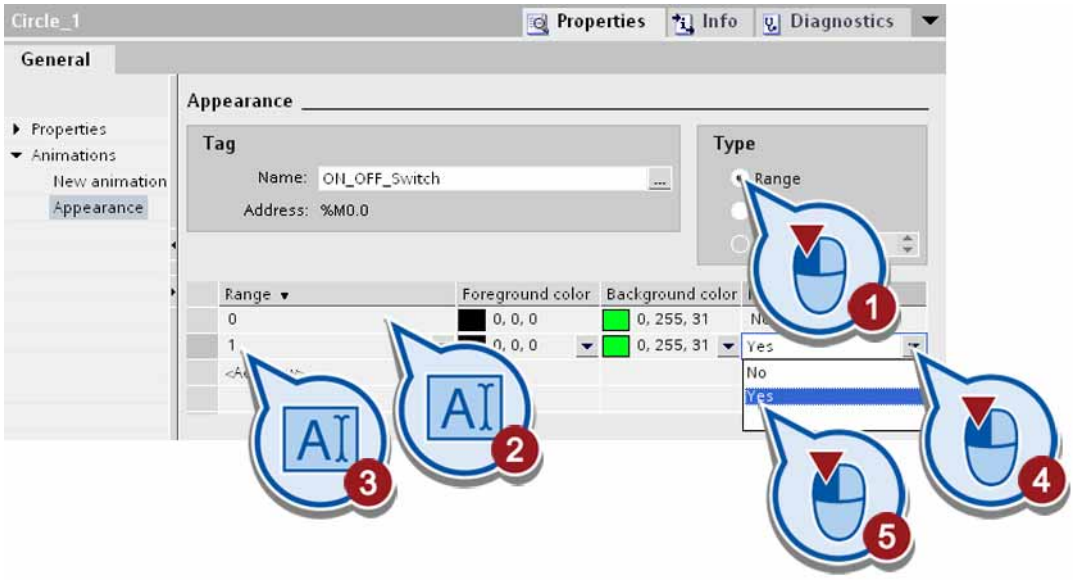
4. For the green LED, create a new animation of the "Appearance" type.



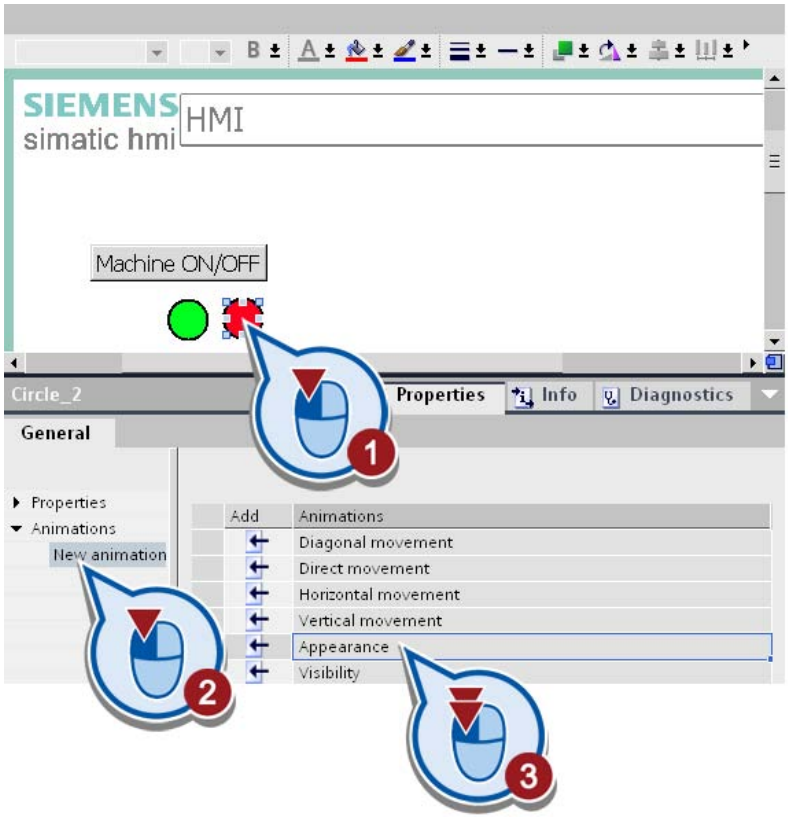
5. Link the animation to the PLC tag "ON_OFF_Switch".



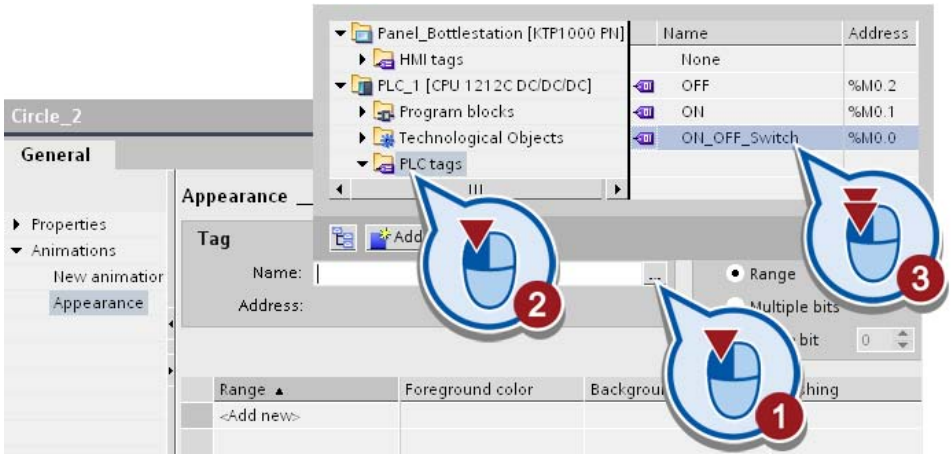
6. Change the appearance of the LED to reflect the status of the PLC tag. The LED should flash as soon as the PLC tag is set to the bit value "1" by the control program.



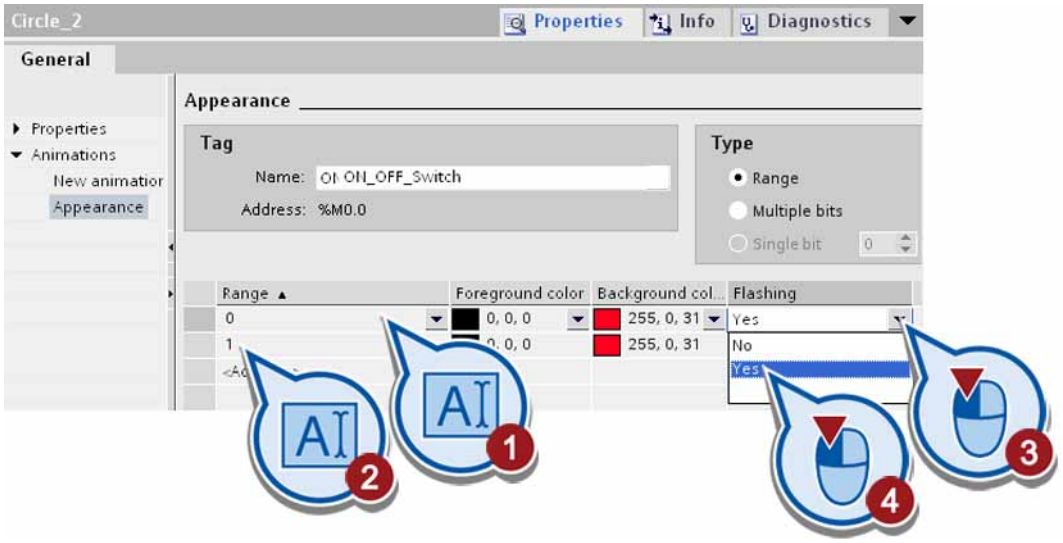
7. For the red LED, create a new animation of the "Appearance" type.



8. You also link this animation to the PLC tag "ON_OFF_Switch".



9. Change the appearance of the LED to reflect the status of the PLC tag. The LED should flash as soon as the PLC tag is set to the bit value "0" by the control program.



Result

You have created and animated status LEDs using the graphic object "circle". The red LED flashes in the initial state.

- When the control program is started by means of the "Machine ON/OFF" button, the bit value of the tag "ON_OFF_Switch" is set to "1" and the green LED flashes.
- When the control program is stopped by pressing the "Machine ON/OFF" button again, the bit value of the tag "ON_OFF_Switch" is set to "0" and the red LED flashes.

In the next section you will create the graphic object "Conveyor".

2.6.4.3 Graphic object "Conveyor"

Introduction

The following steps show you how to link a logic operation to a graphic folder to import graphic objects. You import the graphic object "Conveyor" (Conveyor.Simple.wmf) by means of the logic operation.

Requirements

The HMI screen is open.

Procedure

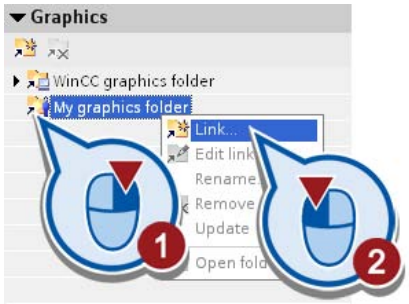
To import a graphic object, follow these steps:

1. Copy the ZIP file "WinCC Graphics" from the following Internet addresss to your local hard disk and extract the file.

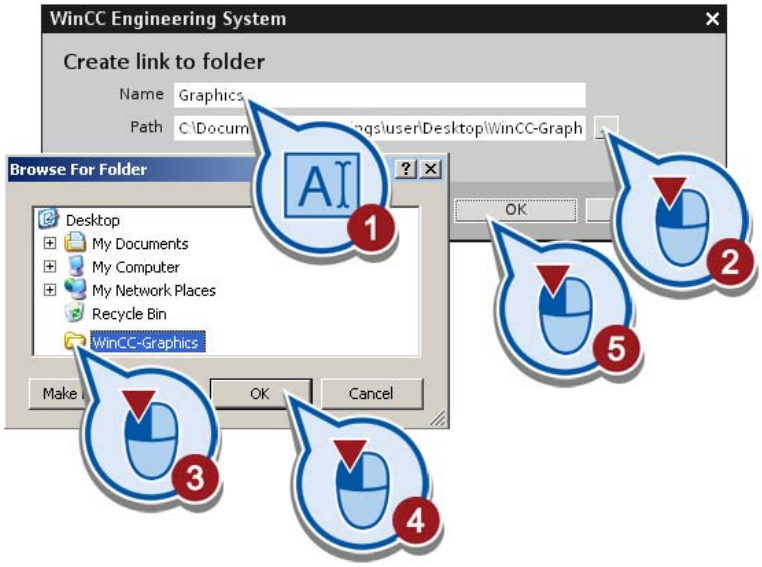
   http://support.automation.siemens.com/WW/view/en/40263542

   Click on "Info" to see the ZIP files.

2. In the "Toolbox" task card open the "Graphics" pane and create a new logic operation.

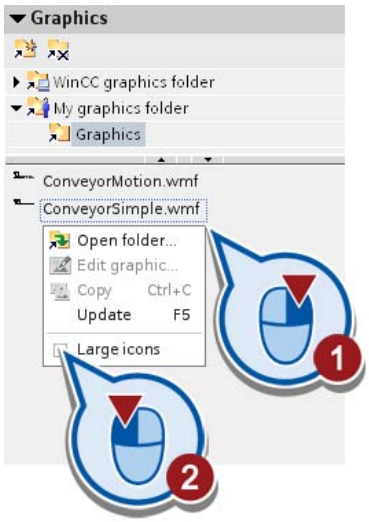


3. Assign a name for the logic operation and select the previously extracted folder "WinCC Graphics".

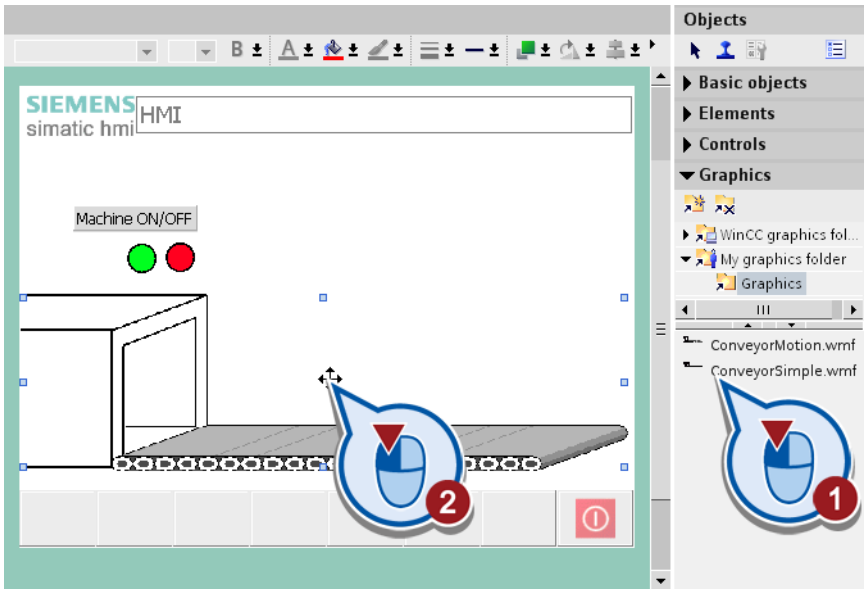


Two graphics are displayed below the newly created logic operation.

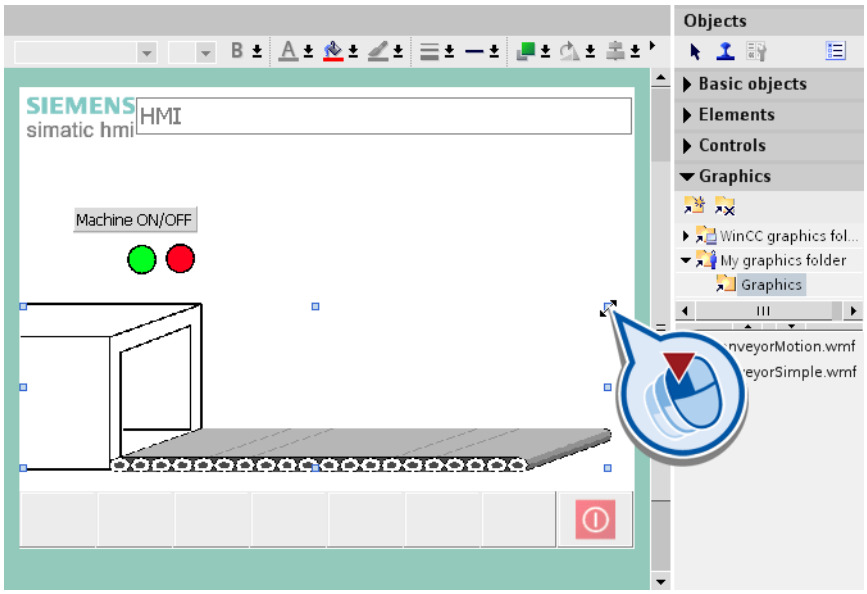
4. Disable the "Large icons" option.



5. Position the graphic object "ConveyorSimple.wmf" in the HMI screen.



6. Scale the graphic object.



Result

You have copied the static graphic object "Conveyor" into the project. If you move or delete the folder "WinCC Graphics", only the shortcut will be lost. The graphic object is maintained in the project.

In the next section you will create the graphic object "Bottle" with a movement animation.

2.6.4.4 Graphic object "Bottle" with motion simulation

Introduction

The following steps show you how to create the graphic object "Bottle" with a movement animation. In the animation the milk bottle moves from left to right across the conveyor. Use an internal HMI tag to animate the objects.

Internal HMI tags

Internal HMI tags have no connection to the PLC. They are stored in the memory of the HMI device. Only the HMI device has read and write access to these tags. You create internal HMI tags, for example, to conduct local computations independently of the control program.

Requirements

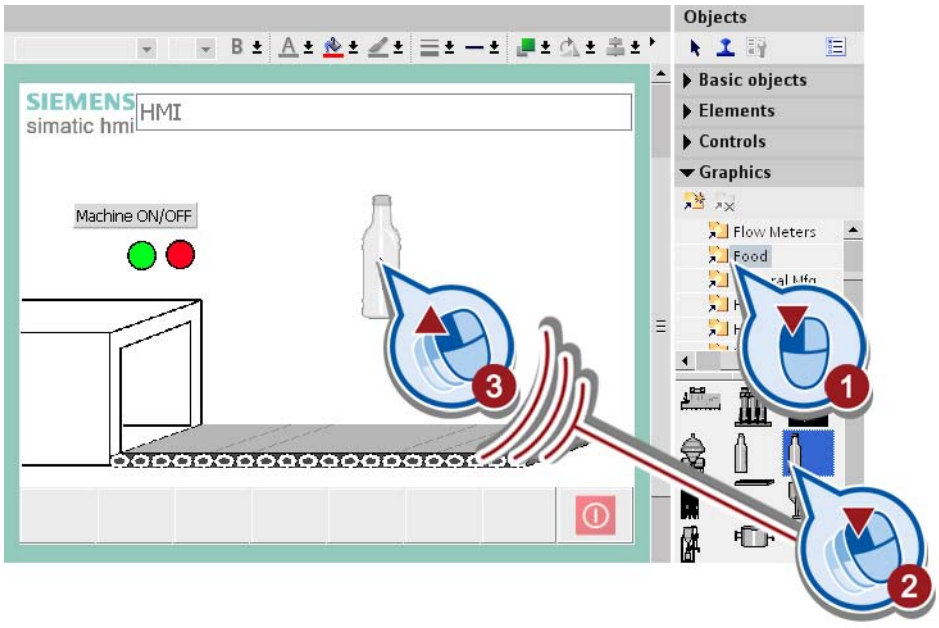
The HMI screen is open.

Procedure

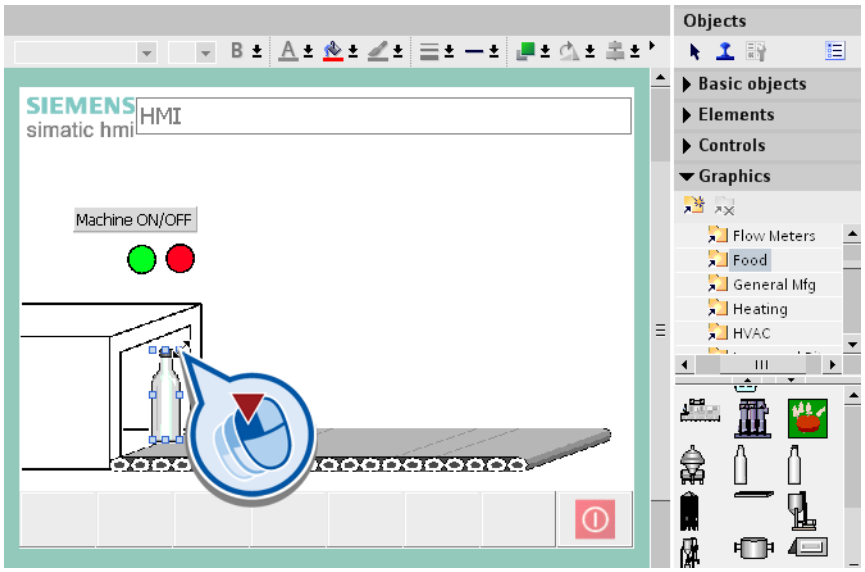
To create the graphic object "Bottle" and to configure a horizontal movement, follow these steps:

1. Copy the graphic object "Bottle" with a drag-and-drop operation from the WinCC graphic folder "Symbol Factory Graphics > SymbolFactory 256 Colors > "Food" to the free area of the screen above the "Conveyor" object.

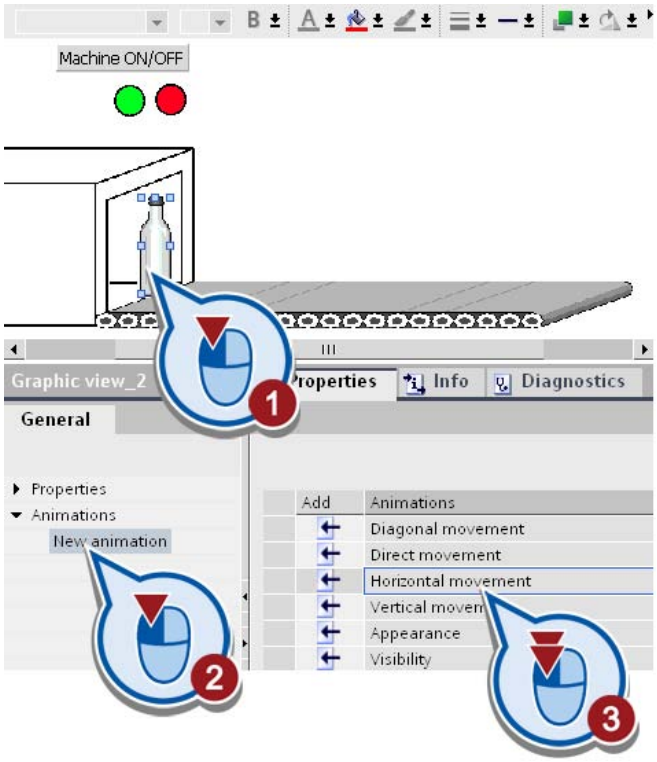
   When you position the bottle, make sure that you put it in a free area of the HMI screen. If you drag the bottle directly onto the conveyor, it will be replaced by the representation of the bottle.



2. Scale the bottle to a height that is lower than that of the shaft.

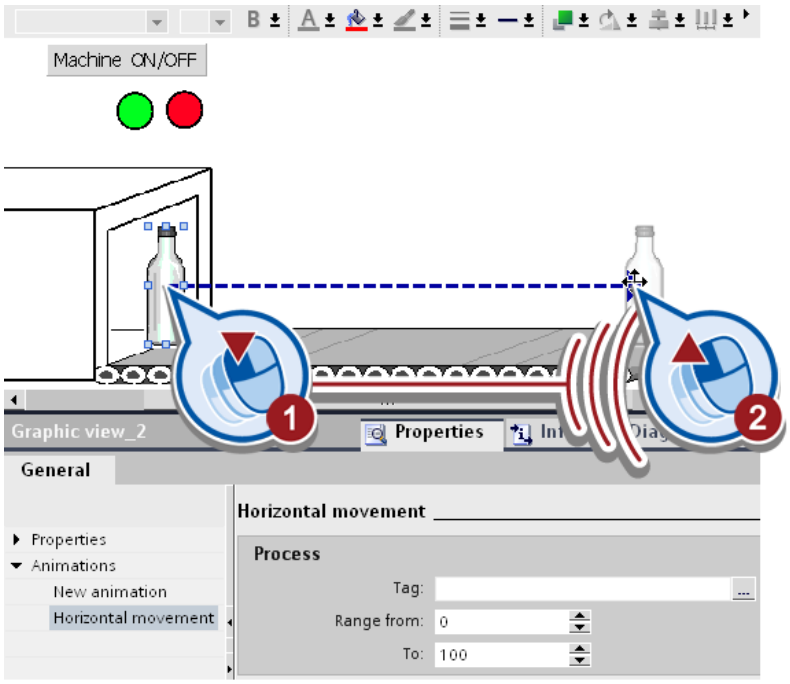


3. Create a horizontal movement animation for the graphic object "Bottle".



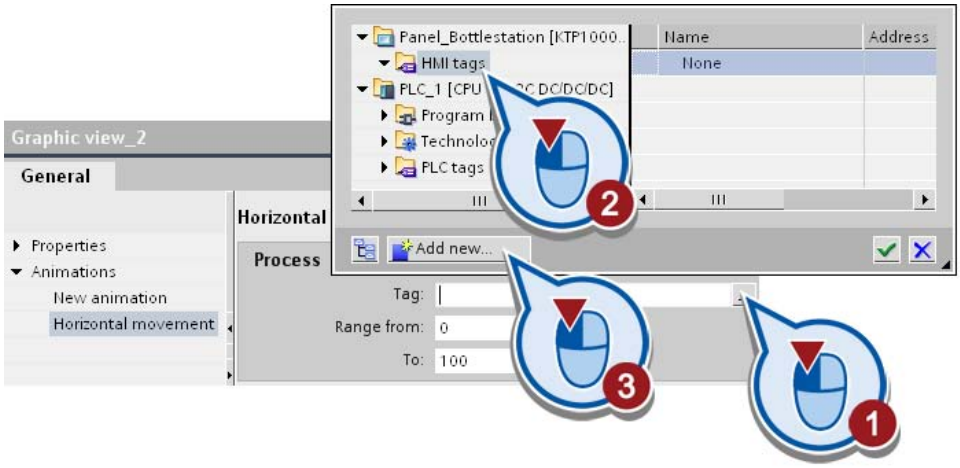
A transparent copy of the bottle is shown in the work area, which is connected to the source object by means of an arrow.

4. Move the transparent bottle to the end of the conveyor.

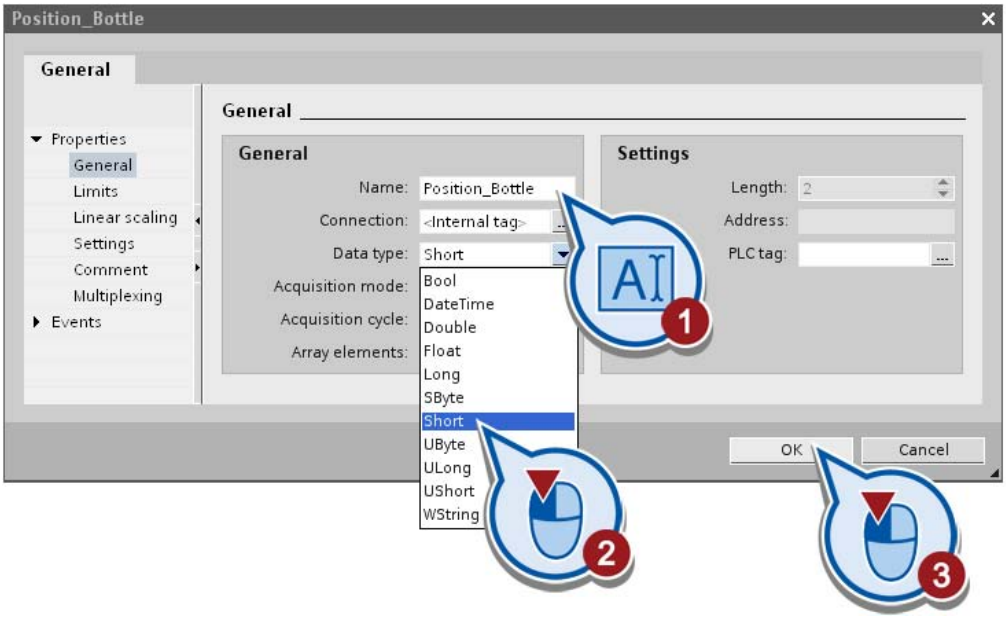


The system automatically enters the pixel values of the final position in the Inspector window.

5. Create a new HMI tag for the movement animation in the Inspector window.



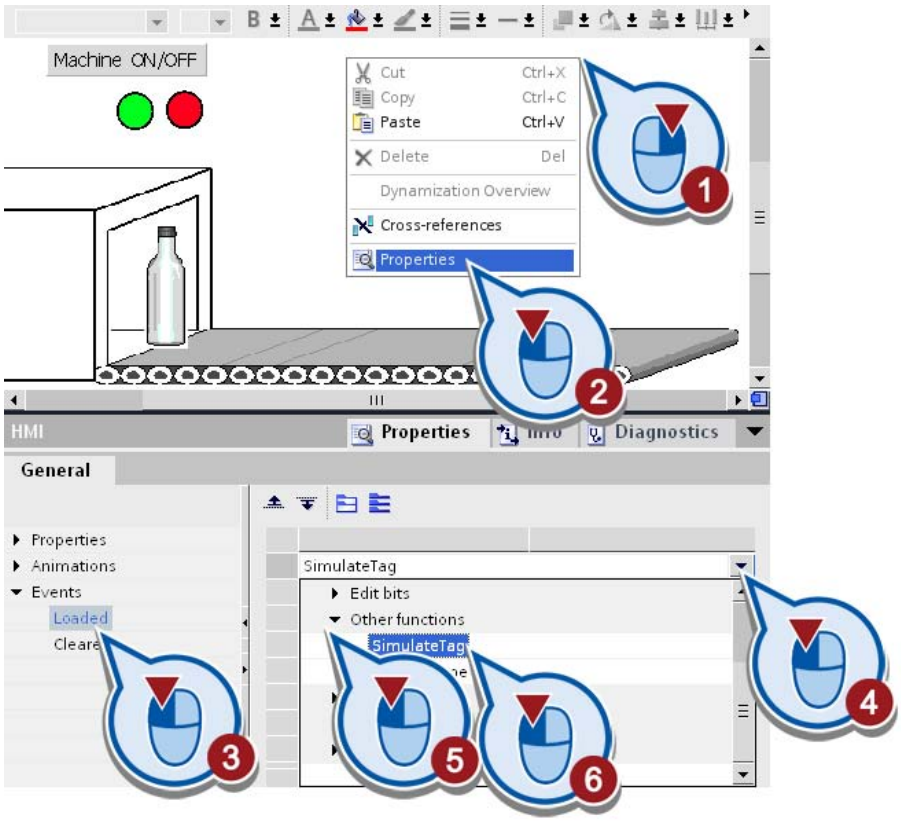
6. Use "Position_Bottle" as name for the tag and "Short" as data type.



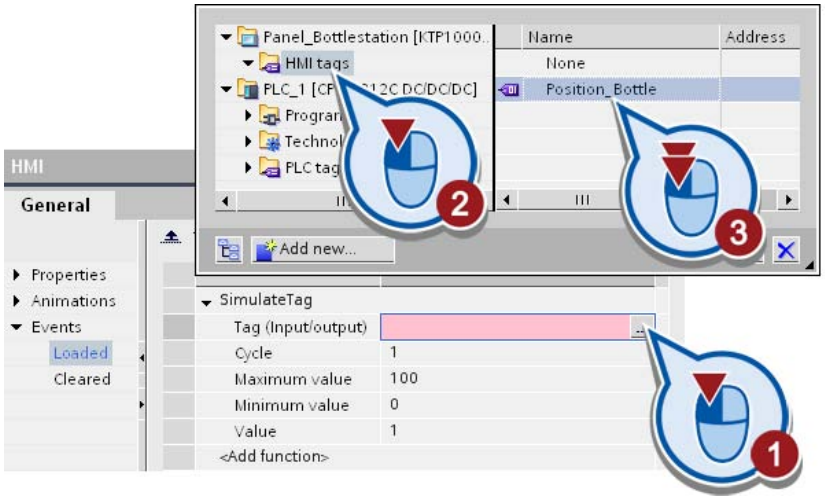
The position of the bottle is linked to the tag. If the value of the tag is changed in the current program, the position of the bottle changes.

To simulate a movement, the value of the tag "Position_Bottle" must change automatically. The value of the tag should increase automatically after loading the HMI screen. As soon as the value 100 is reached, it should start again at 0. The value change for the tag is simulated by means of the properties of the HMI screen.

7. First add the "SimulateTag" function to the event "Loaded" of the HMI screen.



8. Assign the "Position_Bottle" tag to the "Simulate Tag" function.



9. Save the project.

Result

You have created the graphic object "Bottle" with a movement animation. When the HMI screen is loaded to the HMI device, the value of the tag "Position_Bottle" increases by one after each basic cycle (200 ms). As soon the value reaches 100, the value of the tag is set to "0". The position of the bottle is dependent on the value of the tag. If, for example, the tag has the value 50, the bottle is in the middle of the conveyor.

In the next section you will modify the visibility of the bottle by means of the tag "ON_OFF_Switch" of the "Machine ON/OFF" button.

2.6.4.5 Modify visibility of motion animation

Introduction

The movement animation of the bottle is automatically started when the HMI screen is loaded. The follow steps show you how to configure the visibility of the animated bottle in the screen depending on the value of the PLC tag "ON_OFF_Switch".

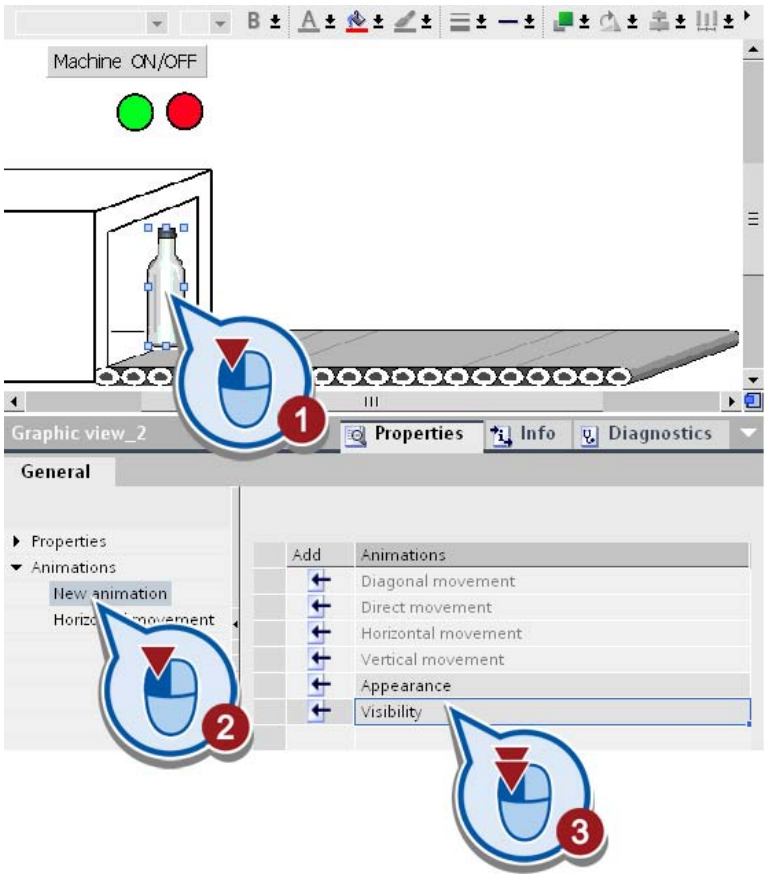
Requirements

The HMI screen is open.

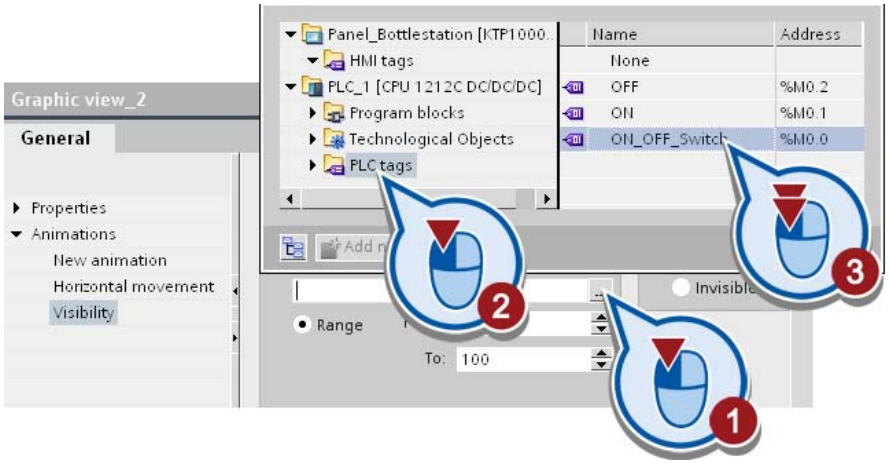
Procedure

To configure the visibility of the bottle in the HMI screen, follow these steps:

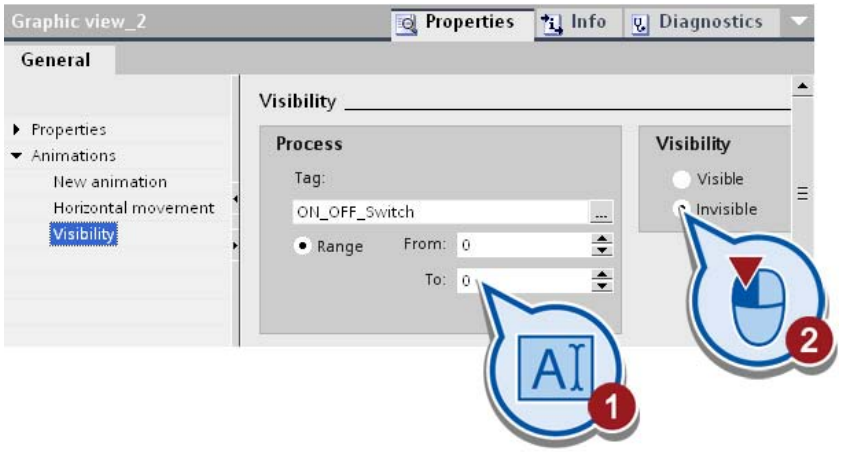
1. Create a new animation, "Visibility", for the graphic object "Bottle".



2. Assign the PLC tag "ON_OFF_Switch" to the animation.



3. Switch the visibility for the tag range "0" to "0" to "Invisible".



4. Click the "Save project" button on the toolbar to save the project.

Result

You have configured the visibility of the bottle in the screen. When the machine is switched on and the process value of the tag "ON_OFF_Switch" has the value "1", the bottle is visible.

2.7 Testing HMI screen

2.7.1 Load HMI screen to the HMI device

Introduction

You can load the Getting Started project to an HMI device and execute it in runtime. To do this, you must have a connection between the configuration device and the HMI device. If you are not using an HMI device, you can simulate the runtime in the TIA Portal (see Simulate runtime (Page 87)).

Runtime in HMI device

HMI devices are used to operate and monitor tasks in process and production automation.

If you are using an HMI device for the Getting Started project, make sure that a connection has been established between the HMI device and the PLC.

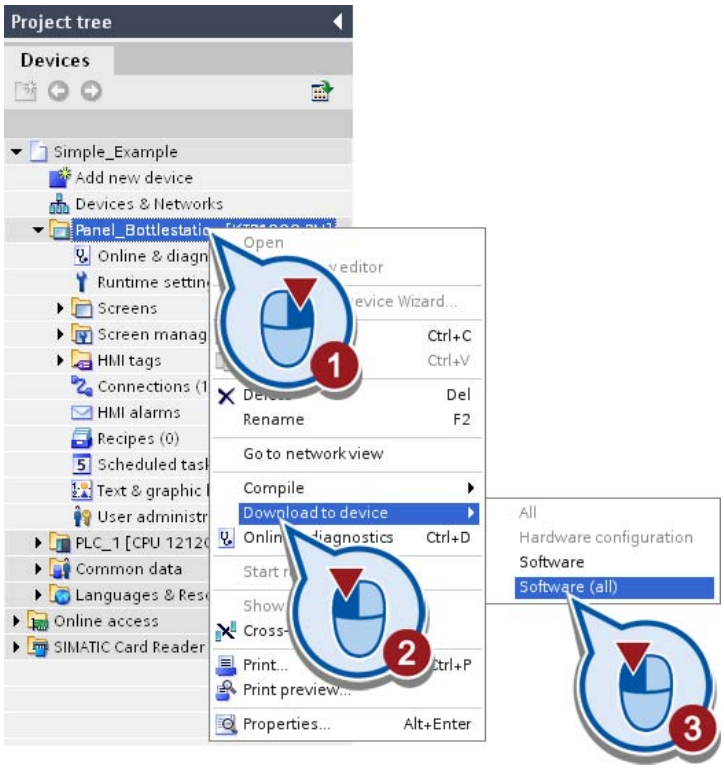
Because the HMI screen of the Getting Started project mainly uses PLC tags, the animations of the objects will only be executed if a connection is established between the HMI device and the PLC.

Requirements

- A connection has been established with the HMI device.
- The HMI device is configured correctly.
- The HMI device is in transfer mode.

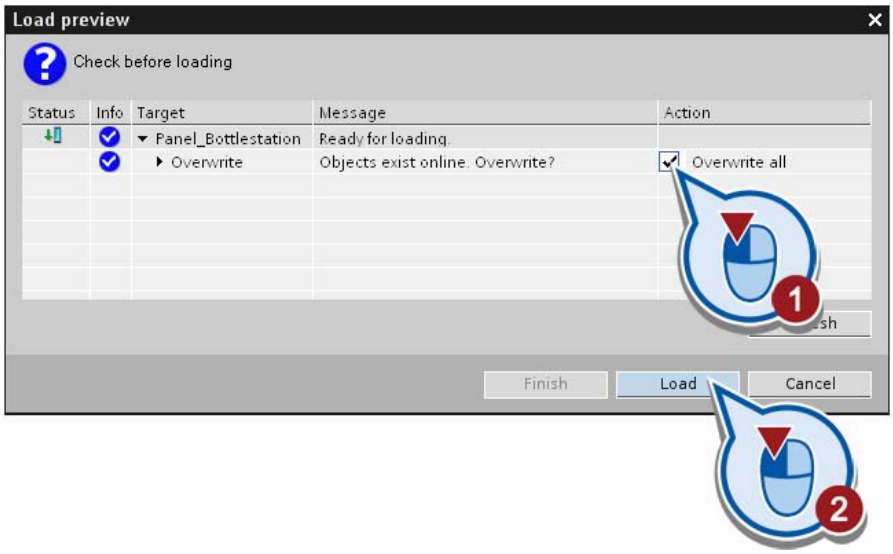
Procedure

1. Start the load process for the software to the HMI device. The project is automatically compiled before the load process.



CAUTION
Make sure that no relevant data are already loaded to the HMI device. Software that was already loaded will be overwritten.

2. You may overwrite software that was previously loaded to the HMI device.



Note

Information on the topic HMI

For more information on HMI and how to configure HMI devices, see the information system of the TIA Portal and your HMI device manual.

2.7.2 Simulate runtime

Introduction

If you are not using an HMI device, you can simulate all used PLC tags with the Runtime Simulator.

Note

Start the runtime simulation with the tag simulator

If you start the simulation without the tag simulator, the buttons and control elements will not be active.

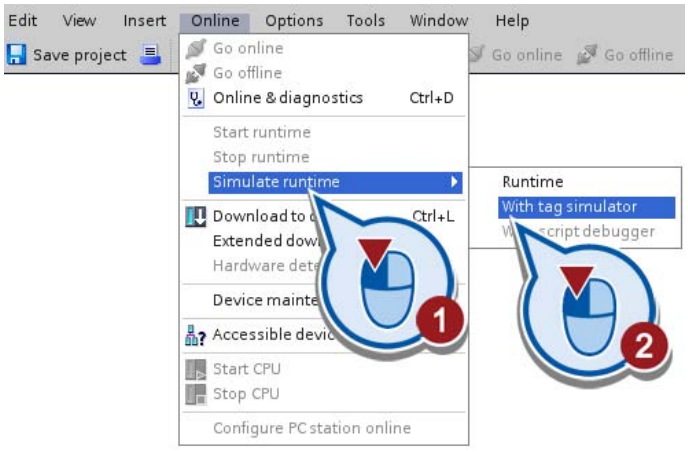
Runtime Simulator

Use the Runtime Simulator to simulate the process values of the connected PLC tags independently of the program. Use the table of the Runtime Simulator to select the PLC tags and to modify their values. The objects in the HMI screen react as though the tags were set by the program of the PLC in runtime.

Procedure

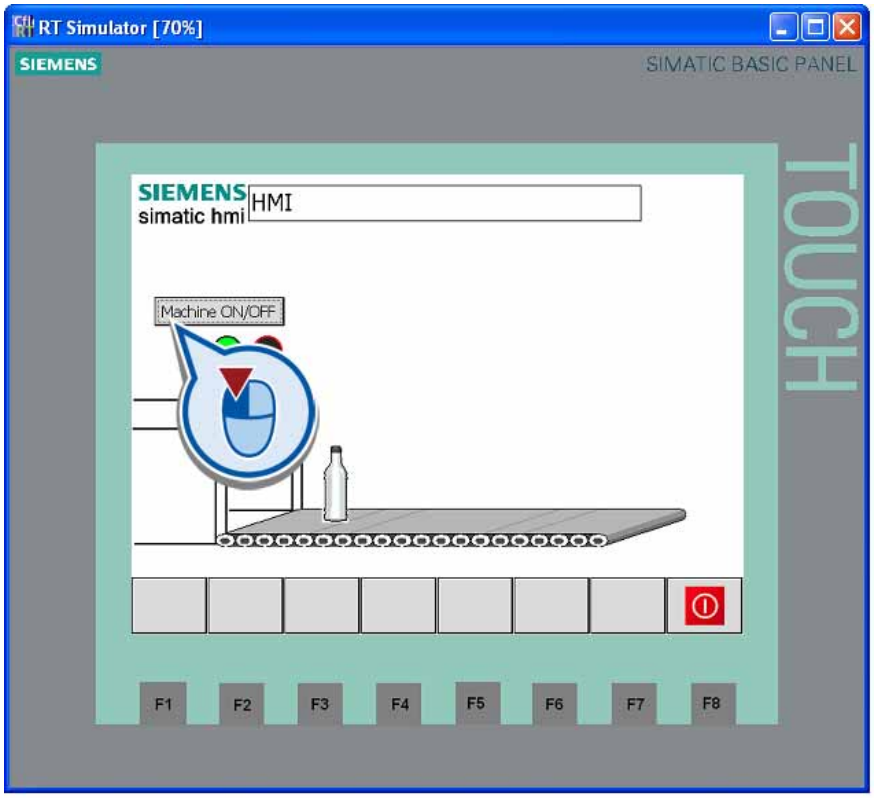
To start the simulation of the HMI screen, follow these steps:

1. Start the runtime simulation via the menu bar. The HMI window must be active. If the menu is inactive, first click on a free area within the HMI screen.



Runtime simulation is started. After the start of the simulation the HMI screen is displayed in the "RT Simulator" window and the red LED flashes (machine switched off).

2. Start the machine.



Note

Error messages during compilation

If the runtime simulation fails to start because of errors in the project, the corresponding error messages are displayed in the Inspector window under "Info > Compile". If you double-click on an error message you will be navigated automatically to the incorrectly configured HMI object.

Result

The movement simulation is started. At the same time the green LED flashes instead of the red LED. If you click the "Machine ON/OFF" button once again, the bottle is no longer visible and the red LED flashes instead of the green LED. To exit the runtime simulation, close the window or click on the "Exit runtime" button.

Extended example

3

3.1 Introduction

Loading a project

If you have skipped the previous chapter, you can load the project status at the end of the chapter (see "Loading a project (Page 17)"). The project status at the end of this chapter is stored in the "Simple_Example.ZIP" file.

Overview

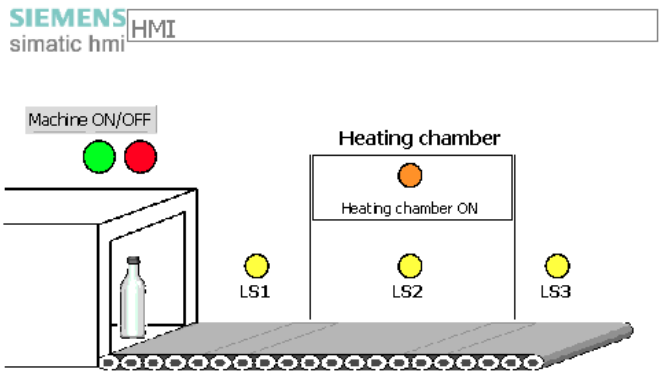
In the second part of the example project you create an additional program that controls a simple station for pasteurizing milk. The station consists of a conveyor and a heating chamber.

The conveyor drive is controlled by the three light barriers:

- The first light barrier (LS1) is activated when a milk bottle stands at the start of the conveyor. The conveyor drive is switched on and the bottle is moved into the heating chamber.
- The second light barrier (LS2) detects the position of a milk bottle in the heating chamber. It signals that the conveyor has to be stopped and for the heating to be switched on in the chamber.

  In the heating chamber the bottles are heated for 40 seconds at a temperature of 75° C. A status light stays illuminated as long as the heating is on. When the heating process is complete, the conveyor drive is restarted and the bottle is transported to the end of the conveyor.
- When a bottle reaches the end of the conveyor a third light barrier (LS3) is activated and the conveyor stops. The bottle can then be collected by the next station.

The switching on and off of the complete station is controlled by the button that was programmed in the first part of the project.



Steps

The following work steps are planned for the second part of the project:

- Expanding the program

  You expand the program by four additional networks.

- Loading the project to the PLC

  You load the expanded project to the PLC and monitor the execution of the program in the online view.

- Test program with program status

  You test the execution of the program with the program status.

- Expanding an HMI screen

  To visualize the program expansion, you insert additional graphic objects into the existing HMI screen.

- Simulating an HMI screen

  You execute the HMI screen in runtime mode.

3.2 Expanding the program

3.2.1 Declaring tags in the PLC tag table

Introduction

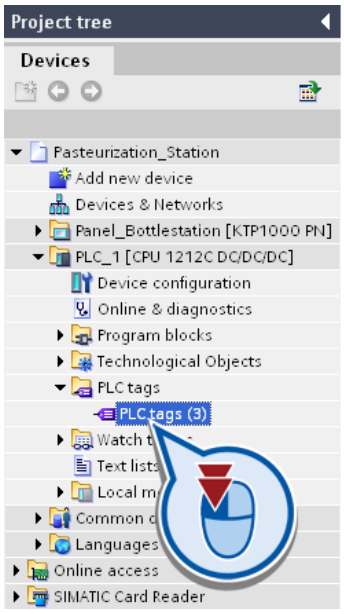
The following steps show you how to define the tags that you need to expand the program. You create the tags in the PLC tag table.

Basic information on the PLC tags and the PLC tag table is available in the "What are tags? (Page 42)" section.

Procedure

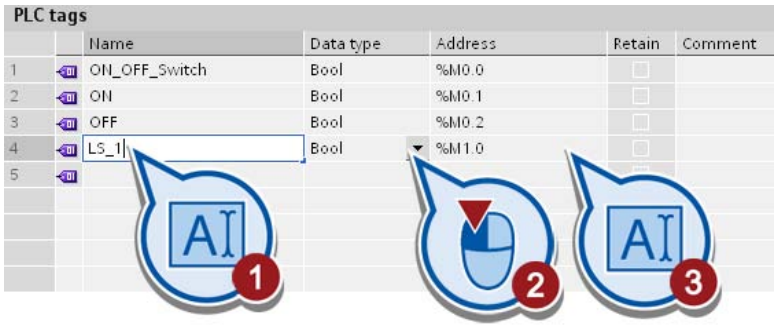
To define the required tags, follow these steps:

1. Open the PLC tag table.

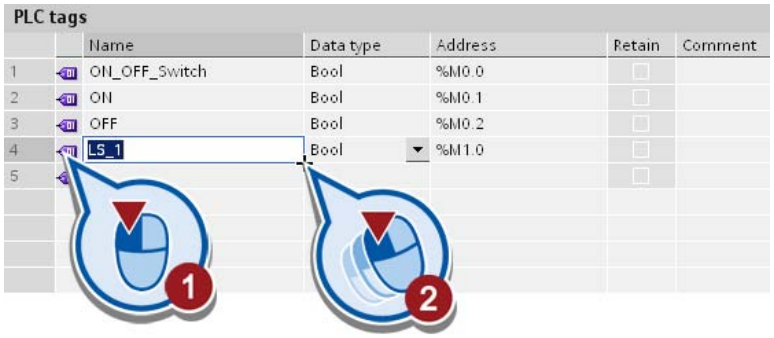


The PLC tag table opens in the work area. The tags defined in the first part of the project are listed.

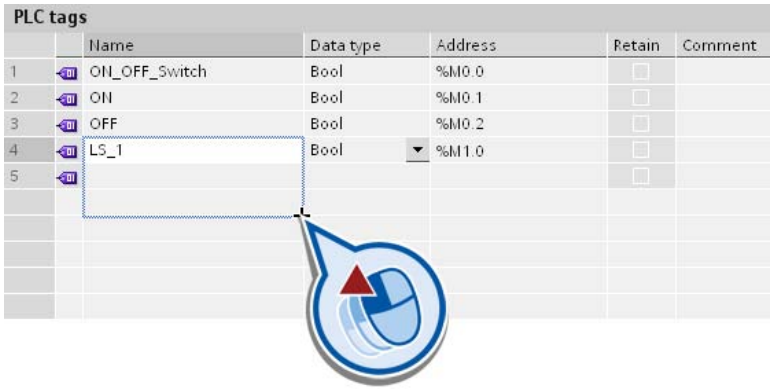
2. Define the tag "LS_1" with the address "M1.0".



3. Select the "Name" cell of the "LS_1" tag and click on the pad character in the bottom right-hand corner of the cell.

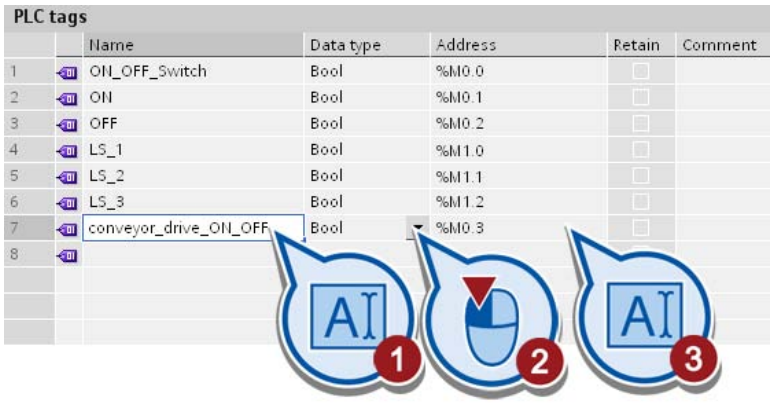


4. Transfer the content of the cell to the two cells below.

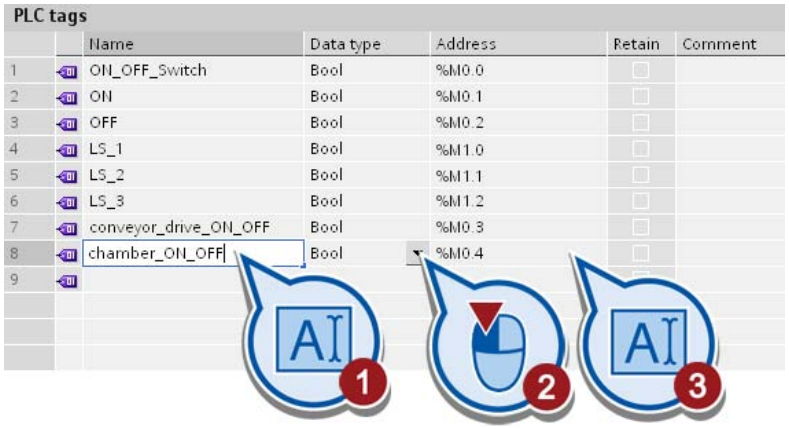


The content is transferred with a consecutive numbering.

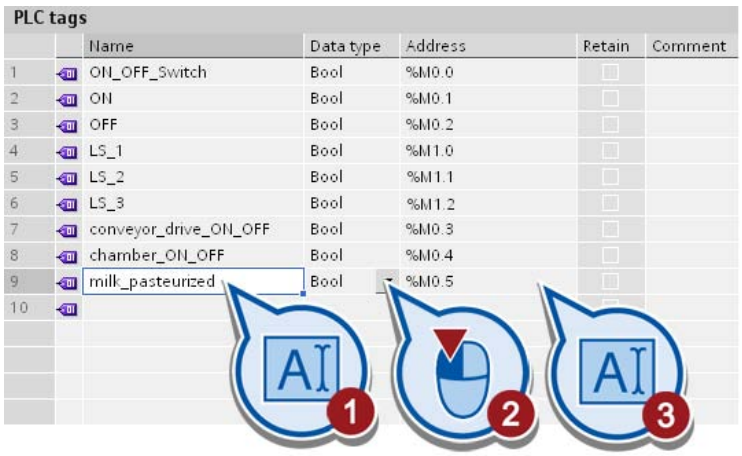
5. Define the "conveyor_drive_ON_OFF" tag.



6. Define the "chamber_ON_OFF" tag.



7. Define the "milk_pasteurized" tag.



Result

You have declared the tags that you need to expand the program. The defined tags are listed in the PLC tag table.

The following table provides an overview of all previously defined tags and the meaning of the respective tag values:

Name	Data type	Address	Function	Meaning of the tag values
ON_OFF_Switch	BOOL	M0.0	Pushbutton switch	• 1: The machine is switched on. • 0: The machine is switched off.
ON	BOOL	M0.1	Switching on the machine	• 1: The machine is switched on. • 0: No effect
OFF	BOOL	M0.2	Switching off the machine	• 1: The machine is switched off. • 0: No effect
LS_1	BOOL	M1.0	Light barrier for detecting the bottle position at the start of the conveyor	• 1: The light barrier 1 is activated. • 0: The light barrier 1 is deactivated.
LS_2	BOOL	M1.1	Light barrier for detecting the bottle position in the heating chamber	• 1: The light barrier 2 is activated. • 0: The light barrier 2 is deactivated.
LS_3	BOOL	M1.2	Light barrier for detecting the bottle position at the end of the conveyor	• 1: The light barrier 3 is activated. • 0: The light barrier 3 is deactivated.
conveyor_drive_ON_OFF	BOOL	M0.3	Operating mode of the conveyor	• 1: The conveyor is put into motion. • 0: The conveyor is stopped.
chamber_ON_OFF	BOOL	M0.4	Operating mode of the heating chamber	• 1: The heating chamber is switched on. • 0: The heating chamber is switched off.
milk_pasteurized	BOOL	M0.5	Status of the pasteurization process	• 1: The milk has been pasteurized. • 0: The milk has not yet been pasteurized.

3.2.2 Programming the conditions for starting the conveyor

3.2.2.1 Querying the status of the machine

Introduction

The following steps show you how to program the status of the machine as condition for starting the conveyor. The first condition is that the conveyor may only be set into motion when the machine is switched on. To implement this condition use a Normally open contact (Page 37), which you interconnect with the tag "ON".

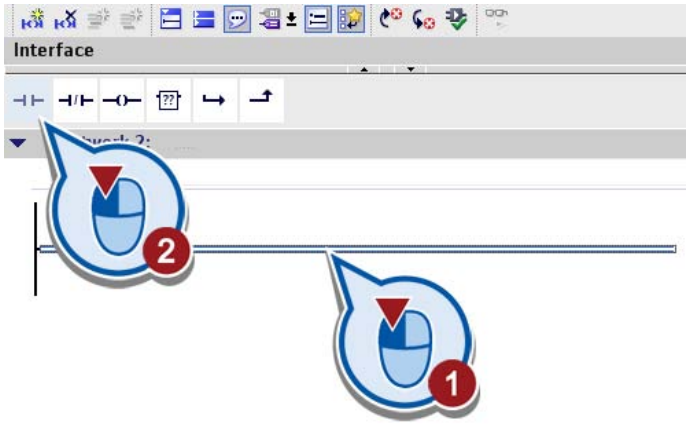
Requirements

The pushbutton switch from the first part of the project has already been programmed in the first network of the organization block "Main [OB1]".

Procedure

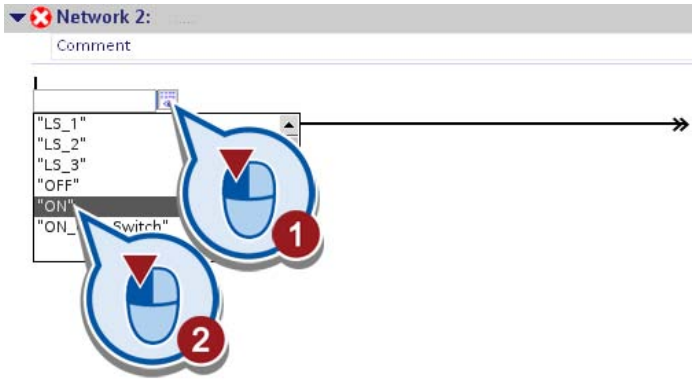
To define the status of the machine as condition for starting the conveyor, follow these steps:

1. Open the organization block "Main [OB1]" in the program editor.
2. Insert a normally open contact into Network 2.



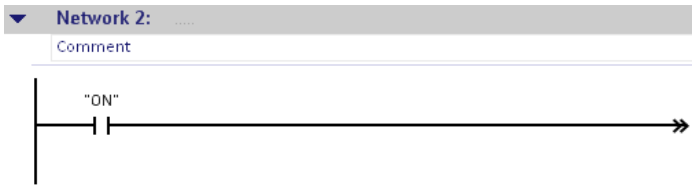
3. Double-click the operand placeholder above the normally open contact.

4. Interconnect the normally open contact with the tag "ON".



Result

You have inserted a normally open contact and interconnected it with the tag "ON".



You have thereby programmed the first condition for starting the conveyor. When the machine is switched on the "ON" tag is set to the signal state "1", the normally open contact closed and the current relayed to the rung. In the next section you will program more conditions, which will be executed depending on the signal state of the "ON" tag.

3.2.2.2 Querying the position of the bottle and the status of the heating chamber

Introduction

The following steps show you how to program the position of the bottle and the status of the heating chamber as condition for starting the conveyor.

You query the status of the heating chamber with the help of the "Normally closed contact" instruction.

Normally closed contact

The following figure shows the icon of the normally closed contact in the program:

<Operand>

---| / |---

The activation of the normally closed contact depends on the signal state of the associated operand. When the operand has signal state "1," the contact is opened and the current flow to the right power rail is interrupted. The output of the instruction in this case results in the signal state "0".

When the operand has signal state "0," the normally closed contact is closed. Power flows through the normally closed contact into the right power rail and the output of the instruction is set to signal state "1".

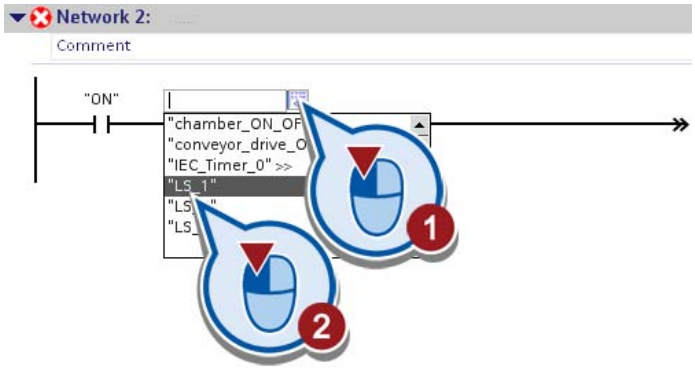
Requirements

- The organization block "Main [OB1]" is open.
- The status of the machine is programmed as first condition for starting the conveyor (see Querying the status of the machine (Page 95))

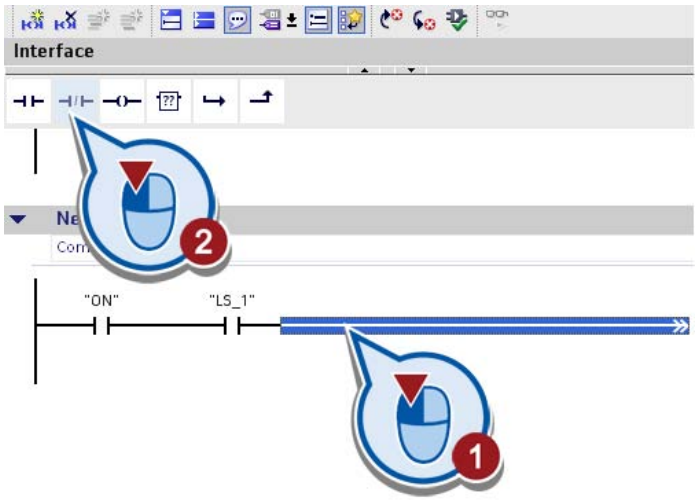
Procedure

To program the position of the bottle and the status of the heating chamber as condition for starting the conveyor, follow these steps:

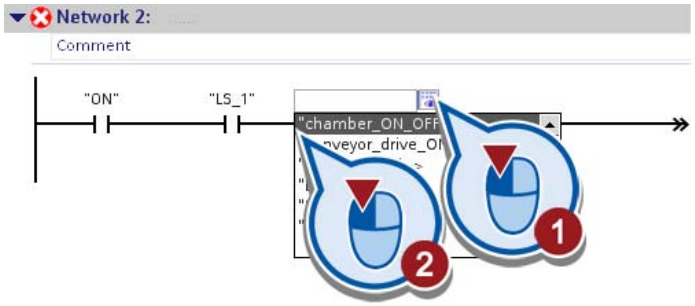
1. Insert a second normally open contact into Network 2.
2. Interconnect the normally open contact with the tag "LS_1".



3. Insert a normally closed contact into Network 2.

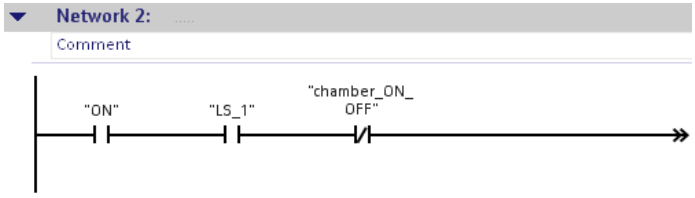


4. Interconnect the normally closed contact with the "chamber_ON_OFF" tag.



Result

You have programmed the position of the bottle and the status of the heating chamber as condition for starting the conveyor.



3.2.2.3 Query pasteurization progress

Introduction

The following steps show you how to control the conveyor depending on the progress of the pasteurization process. When the milk in a bottle has been pasteurized, the conveyor is set into motion and the bottle transported to the end of the conveyor. The information about the progress of the pasteurization process is stored in the "milk_pasteurized" tag and queried with the help of a normally open contact.

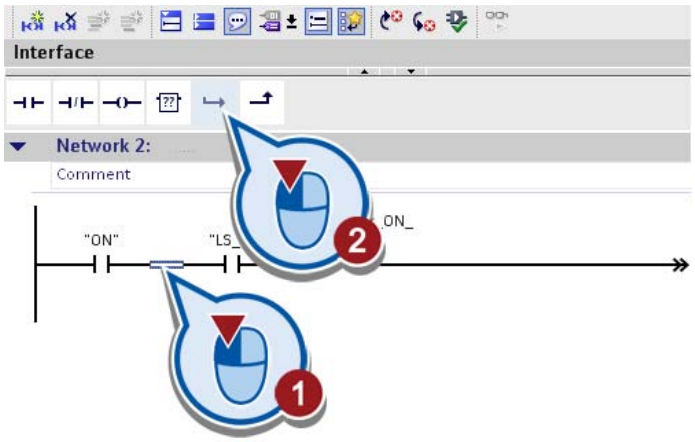
Requirements

- The organization block "Main [OB1]" is open.
- The conditions described in the sections "Querying the status of the machine (Page 95)" and "Querying the position of the bottle and the status of the heating chamber (Page 97)" have already been programmed.

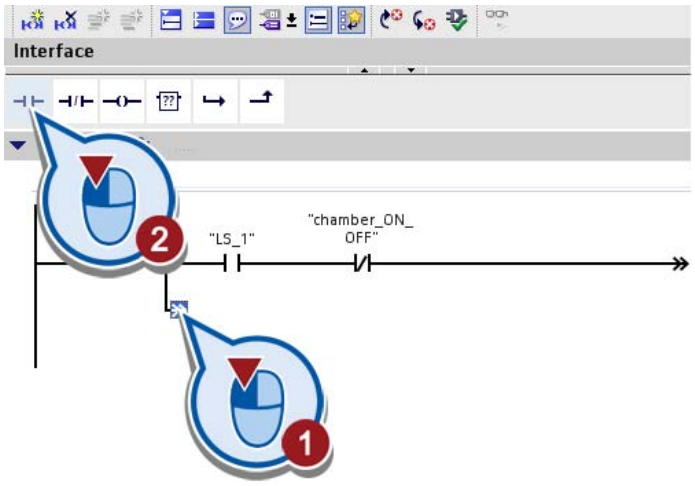
Procedure

To control the conveyor depending on the progress of the pasteurization process, follow these steps:

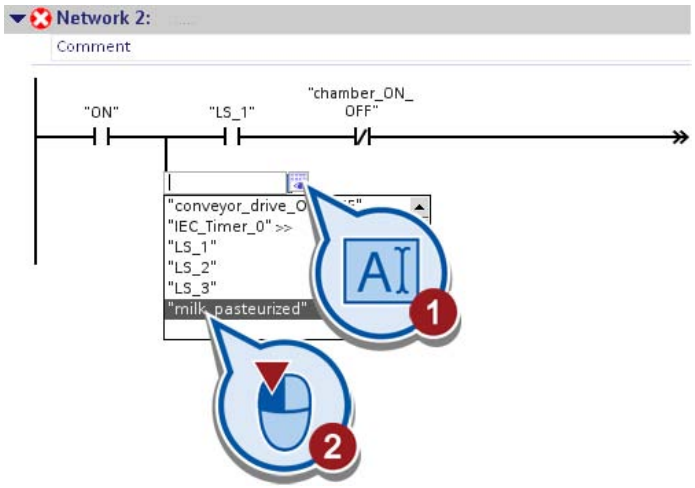
1. Insert a branch in the rung between the first and second normally open contact.



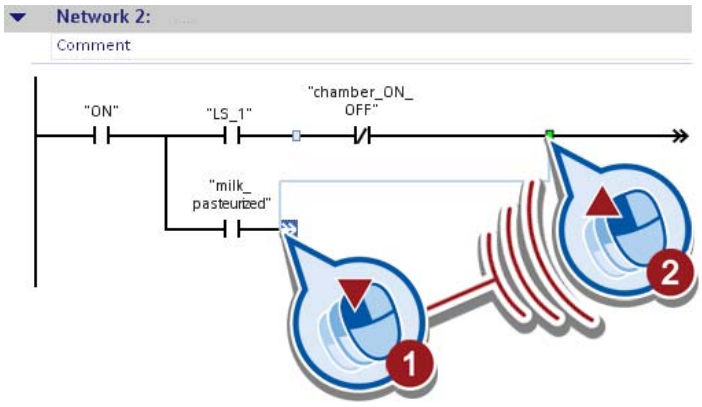
2. Insert a normally open contact in the open branch.



3. Interconnect the normally open contact with the "milk_pasteurized" tag.

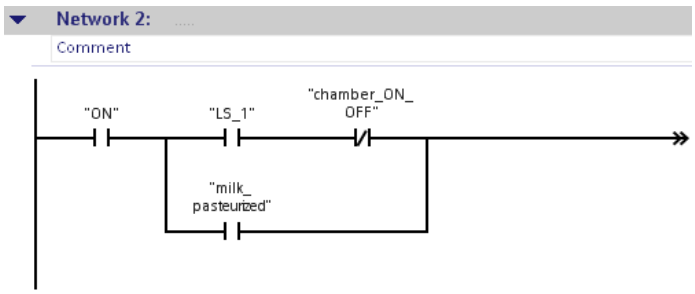


4. Close the open branch.



Result

You have programmed the pasteurization progress as another condition for starting the conveyor.



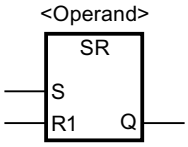
3.2.2.4 Control conveyor

Introduction

The following steps show you how to set the conveyor in motion depending on the programmed conditions. To control the conveyor, use the "Set/reset flip-flop" instruction.

Set/reset flip-flop

The following figure shows the box of the "Set/reset flip-flop" instruction.



You can use this instruction to set or reset the bit of a specified operand depending on the signal state on the S and R1 inputs of the instruction.

- If the signal state is "1" at the S input and "0" at the R1 input, the specified operand will be set to "1".
- If the signal state at the S input is "0" and at the R1 input "1", the specified operand will be reset to "0" because the input R1 has a higher priority than input S.
- When the signal state is "1" on both inputs S and R1, the signal state of the specified operand is reset to "0".

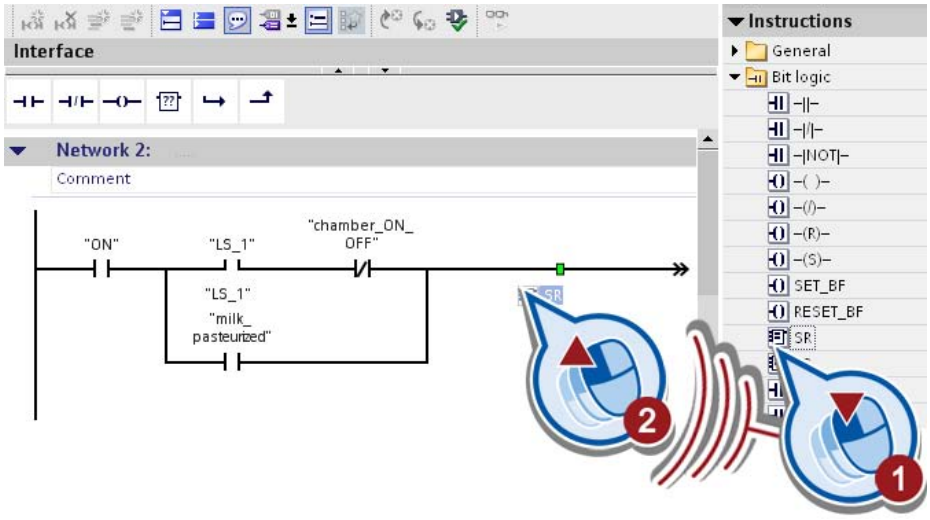
Requirements

- The organization block "Main [OB1]" is open.
- The conditions described in the sections "Querying the status of the machine (Page 95)", "Querying the position of the bottle and the status of the heating chamber (Page 97)" and "Query pasteurization progress (Page 99)" have been programmed.

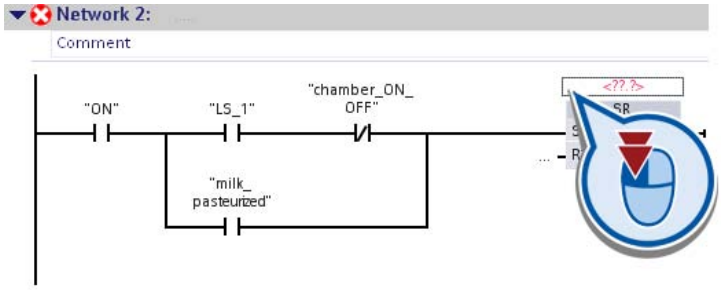
Procedure

To program the control of the conveyor, follow these steps:

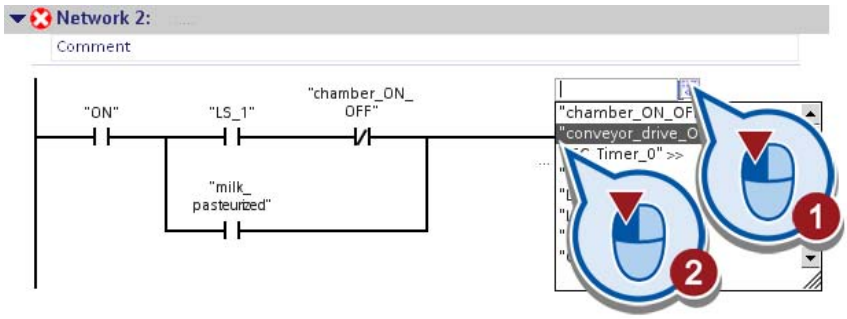
1. Insert the "Set/reset flip-flop" instruction (SR) at the end of the second network.



2. Double-click on the operand placeholder above the instruction.



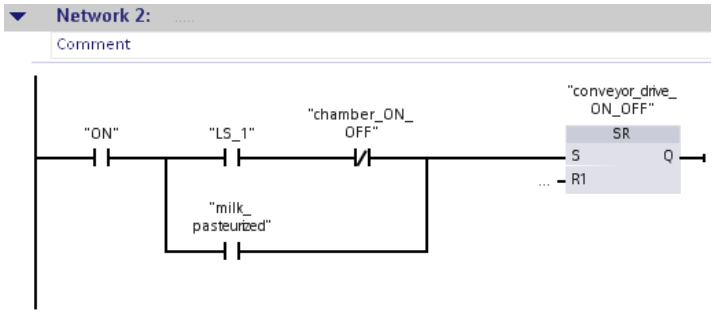
3. Interconnect the instruction with the "conveyor_drive_ON_OFF" tag.



4. Click the "Save project" button on the toolbar to save the project.

Result

You have programmed the conditions for starting the conveyor.



The conveyor is set into motion when the signal state of the "ON" tag is "1" and at least one of the following conditions applies:

- A bottle is detected by the light barrier (LS1) at the start of the conveyor and the heating chamber is switched off. In this case the "LS_1" tag supplies the signal state "1" and the "chamber_ON_OFF" tag the signal state "0".
- The milk bottle has been pasteurized. In this case the "milk_pasteurized" tag has the signal state "1". The bottle is transported to the end of the conveyor.

When the conveyor is set into motion, the "conveyor_drive_ON_OFF" tag has the signal state "1".

In the next section you will program the conditions for stopping the conveyor.

3.2.3 Programming the conditions for stopping the conveyor

Introduction

The following steps show you how to program the conditions for stopping the conveyor. You define these at the input R1 of the "Set/reset flip-flop" instruction, which you have inserted in the last section of the organization block "Main [OB1]".

You program the conditions for stopping the conveyor with the help of the following instructions:

- Normally open contact
- Normally closed contact
- Set/reset flip-flop

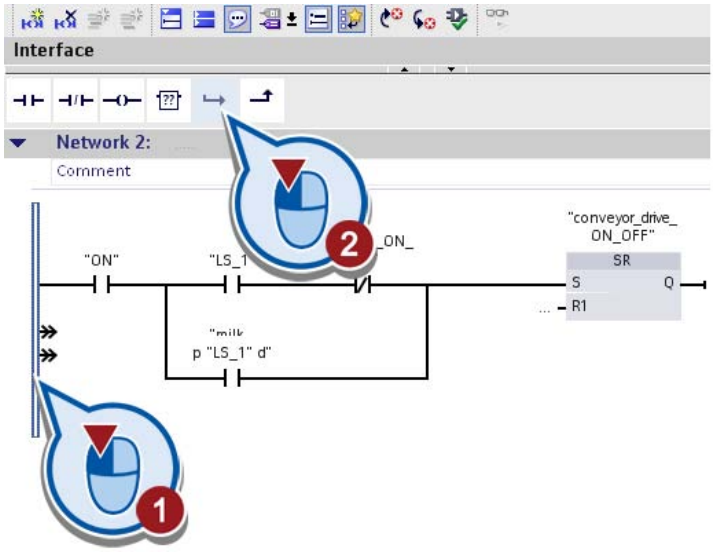
Requirements

- The pushbutton switch from the first part of the project has already been programmed in the first network of the organization block "Main [OB1]".
- The conditions for starting the conveyor have already been programmed in the organization block "Main [OB1]".
  See also: "Programming the conditions for starting the conveyor (Page 95)"

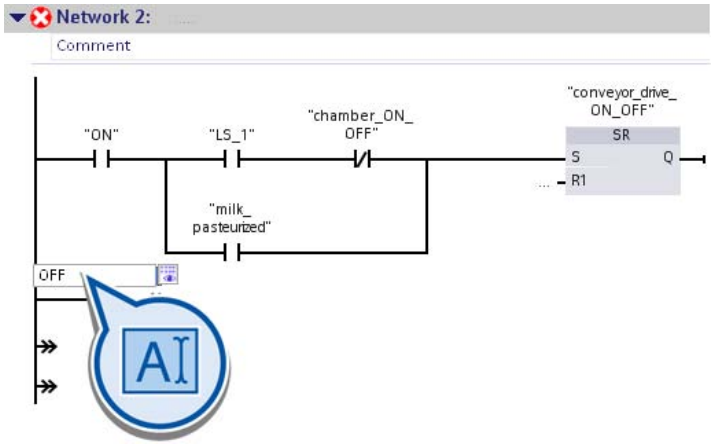
Procedure

To program the conditions for stopping the conveyor, follow these steps:

1. Open Network 2 in the organization block "Main [OB1]".
2. Add three new rungs.



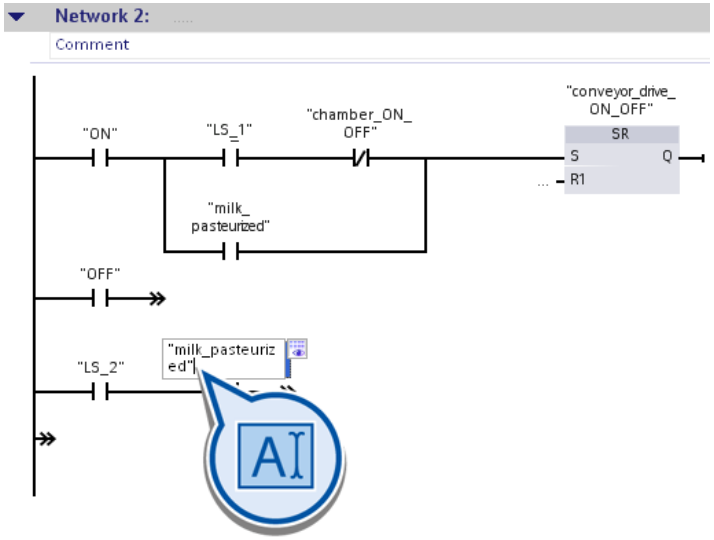
3. Insert a normally open contact into the first new rung and interconnect it with the tag "OFF".



   By interconnecting the tag "OFF" you define the condition that the conveyor will stop when the machine is switched off.

4. Insert a normally open contact into the second new rung and interconnect it with the tag "LS_2".

5. Insert a normally closed contact at the end of the second rung and interconnect it with the tag "milk_pasteurized".

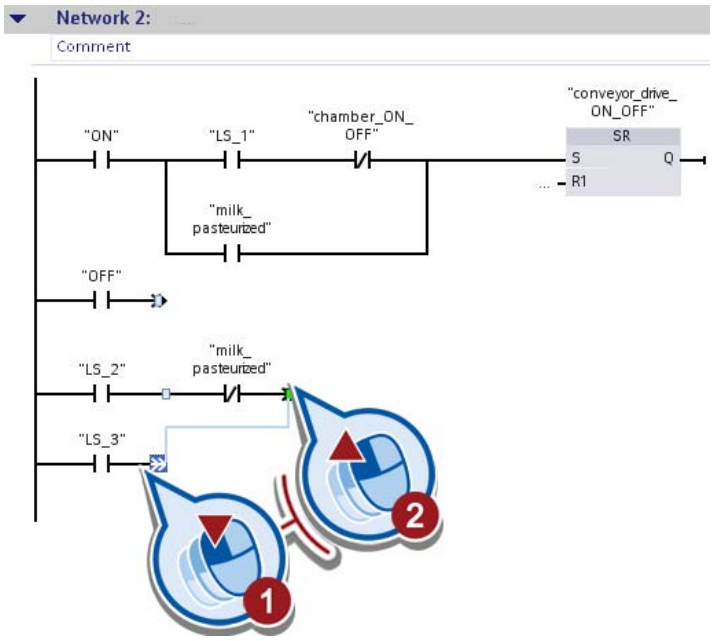


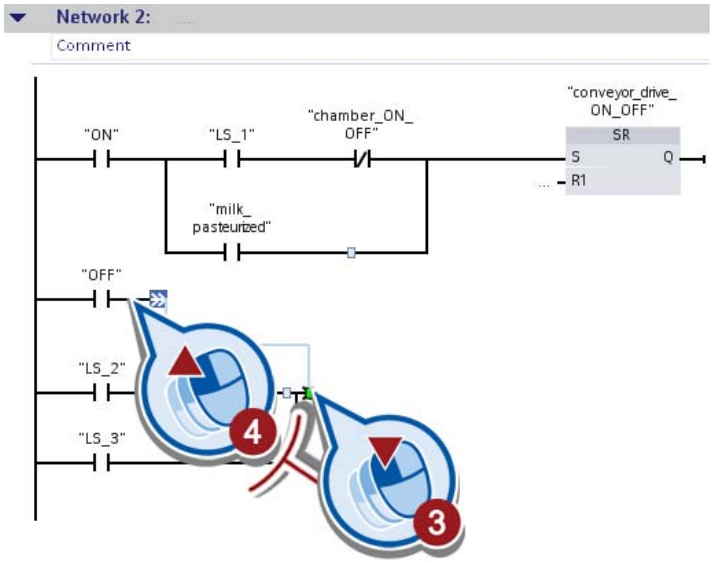
By interconnecting the tags "LS_2" and "milk_pasteurized" you define the condition that the conveyor will be stopped when a bottle has reached the heating chamber and the milk has not been pasteurized yet.

6. Insert a normally open contact into the third new rung and interconnect it with the tag "LS_3".

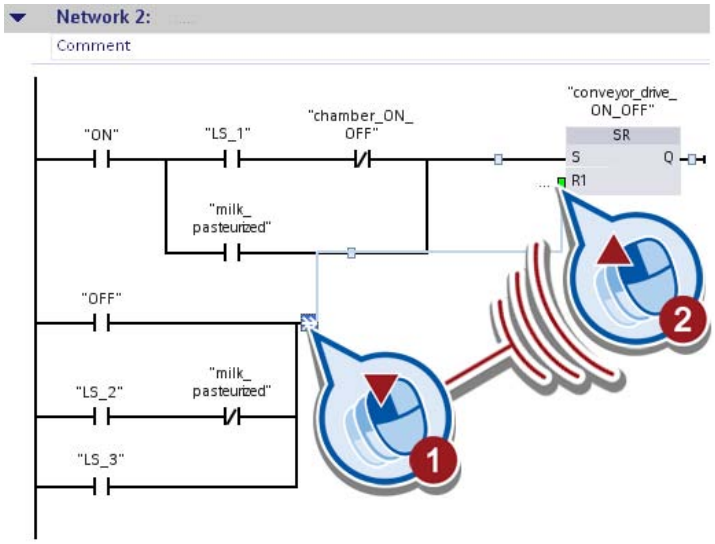
   By interconnecting the tag "LS_3" you define the condition that the conveyor will stop when a bottle has reached the end of the conveyor.

7. Interconnect the inserted normally open contacts in a parallel connection.



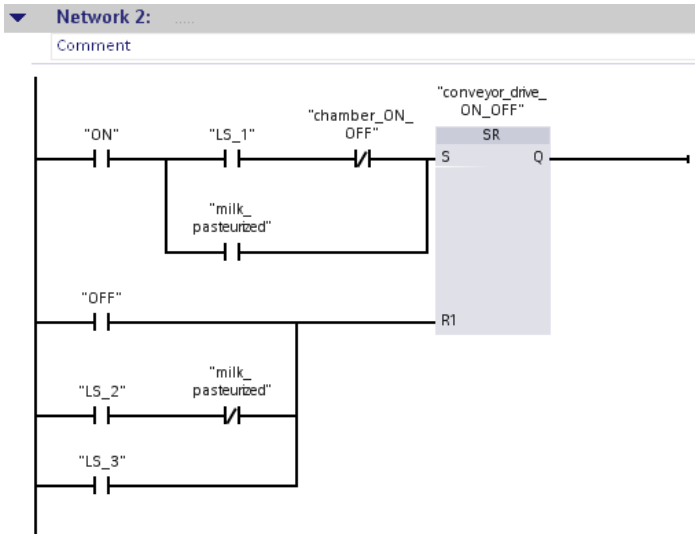


8. Interconnect the parallel connection with the input R1 of the instruction "Set/reset flip-flop".



Result

You have programmed the PLC of the conveyor drive. The conveyor will be driven or stopped depending on which position a milk bottle has on the conveyor. The position of the milk bottles is detected with the help of the light barriers "LS_1", "LS_2" and "LS_3".



The conveyor is stopped when the signal state of the tag "OFF is "1" or at least one of the following conditions applies:

- The bottle was detected by the light barrier (LS2) in the heating chamber and the milk is not yet pasteurized. In this case the "LS_2" tag supplies the signal state "1" and the "milk_pasteurized" tag the signal state "0".
- The bottle was detected by the light barrier (LS3) at the end of the conveyor. In this case the "LS_3" tag has the signal state "1".

When the the conveyor is stopped, the tag "conveyor_drive_ON_OFF" has the signal state "0".

In the next section you will program the switching on and off of the heating chamber.

3.2.4 Programming the PLC of the heating

Introduction

The following steps show you how to program the PLC of the heating process. The heating process is controlled by the switching on and off of the heating chamber. In the example project it will be assumed for the sake of simplicity that the temperature of 75°C has already been reached in the heating chamber when it is switched on.

You program the PLC of the heating chamber in the third network of the organization block "Main [OB1]" with the help of the following instructions:

- Normally open contact
- Set/reset flip-flop

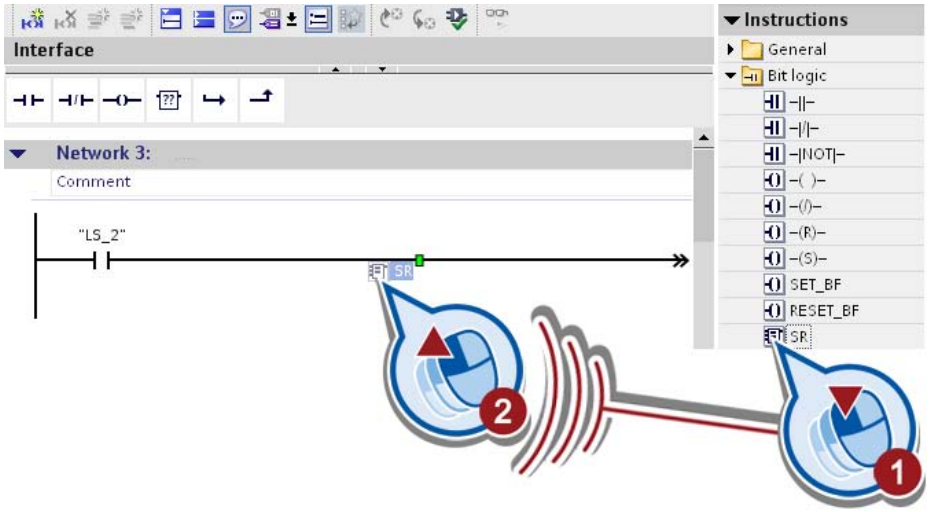
Requirements

- The organization block "Main [OB1]" is open.
- Networks 1 and 2 of the organization block "Main [OB1]" have been programmed.

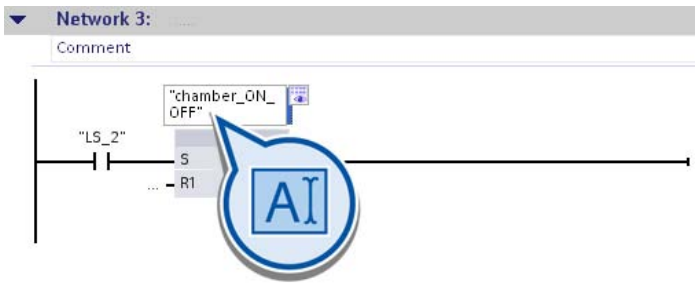
Procedure

To program the PLC of the heating process, follow these steps:

1. Open the third network of the organization block "Main [OB1]".
2. Insert a normally open contact.
3. Interconnect the normally open contact with the tag "LS_2".
4. Insert the instruction "Set/reset flip-flop".

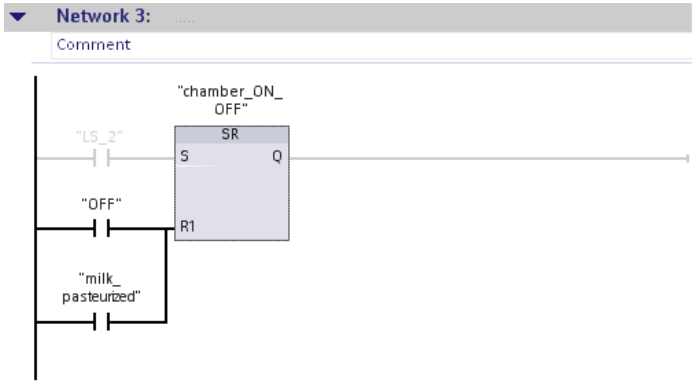


5. In the operand placeholder, click above the instruction "Set/reset flip-flop" and interconnect this with the tag "chamber_ON_OFF".



When the light barrier (LS2) detects a bottle, the tag "LS_2" has the signal state "1" at the input of the instruction "Set/reset flip-flop". The chamber_ON_OFF tag is thereby set and the heating chamber is switched on.

6. Program the following conditions for switching off the heating chamber at the input R1 of the instruction:

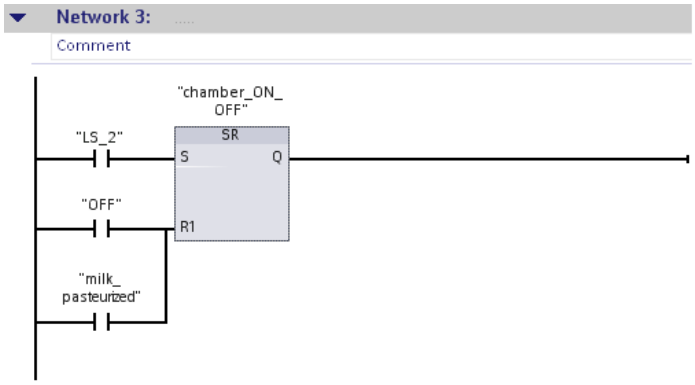


The tag "chamber_ON_OFF" is reset and the heating will be switched off if one of the following conditions applies:

- The machine has been switched off. In this case the "OFF" tag has the signal state "1".
- The milk has been pasteurized. In this case the "milk_pasteurized" tag has the signal state "1".

Result

You have programmed the PLC of the heating process.



The heating chamber is switched on when a milk bottle is detected by the light barrier "LS2". When the specified heating period has expired and the milk is pasteurized the heating chamber is switched off and the bottle will be transported to the end of the conveyor.

In the next section you will program the settings for the heating period.

3.2.5 Programming the heating period

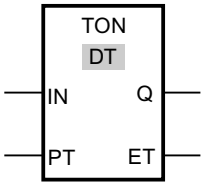
Introduction

The following steps show you how to program the duration of the heating process. To pasteurize the milk the bottles are heated in the heating chamber for 40 seconds. You program the setting for the heating period in the fourth network of the organization block "Main [OB1]" with the help of the following instructions:

- Normally open contact
- On delay
- Output coil

On delay

The following figures shows the box of the "On delay" instruction.



You can use this instruction to set a specified operand with a time delay to the signal state "1". The instruction is executed when the signal state at the input IN of the instruction changes from "0" to "1" (rising edge). The delay period (PT) begins when the instruction starts. When the time expires, the output Q has the signal state "1".

Output coil

The following figure shows the icon of the "Output coil" instruction in the program:

<Operand>

---( )---

You can use the "Output coil" instruction to set the bit of a specified operand. When the result of logic operation (RLO) at the input of the coil is "1," the specified operand is set to signal state "1". When the signal state is "0" at the input of the coil, the bit of the specified operand is reset to "0".

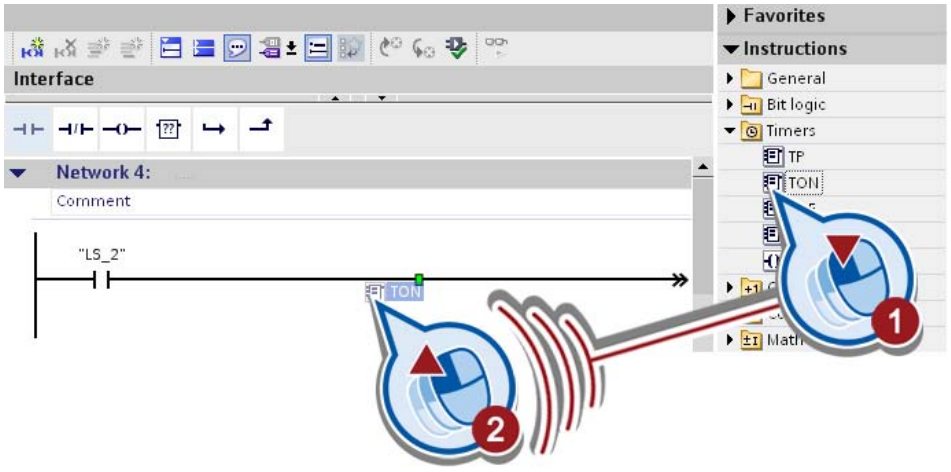
Requirements

- The organization block "Main [OB1]" is open.
- Networks 1 to 3 of the organization block "Main [OB1]" have been programmed.

Procedure

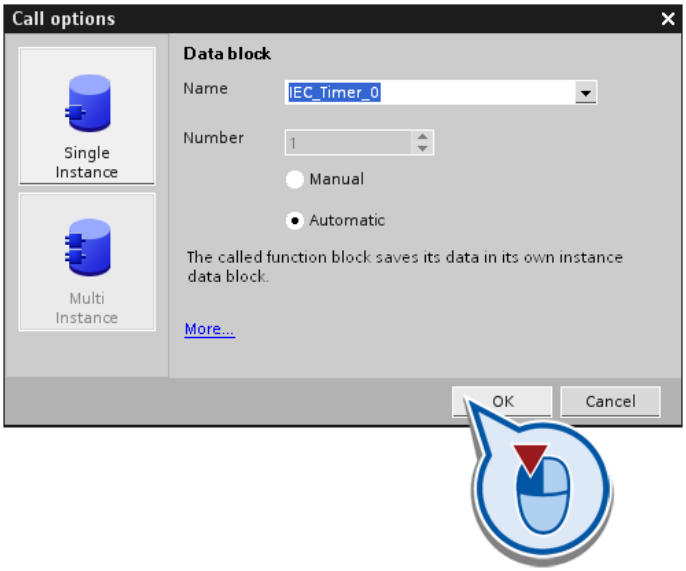
To program the duration of the heating process, follow these steps:

1. Open the fourth network of the organization block "Main [OB1]".
2. Insert a normally open contact.
3. Interconnect the normally open contact with the tag "LS_2".
4. Insert the instruction "On delay" (TON).



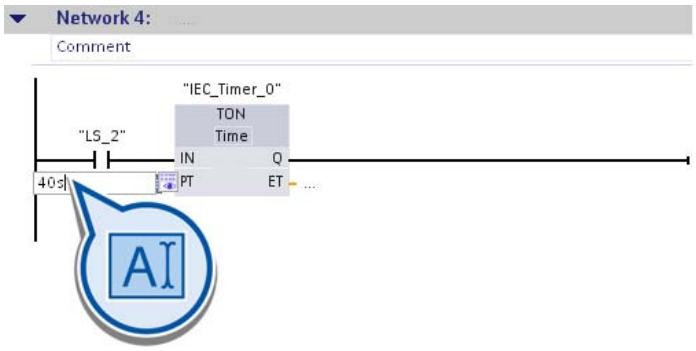
The "Call options" dialog box for creating a data block will open.

5. Create the data block "IEC_Timer_0".



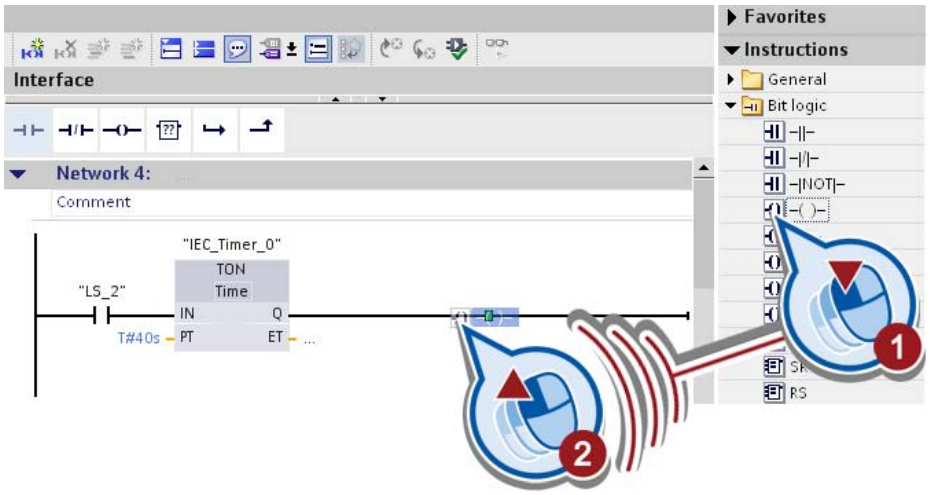
The data block ""IEC_Timer_0" is created in the "Programming blocks" folder and assigned to the instruction "On delay". The data of the inserted time instruction are stored in this data block.

6. Enter the heating period of 40 seconds at the input PT of the time instruction.

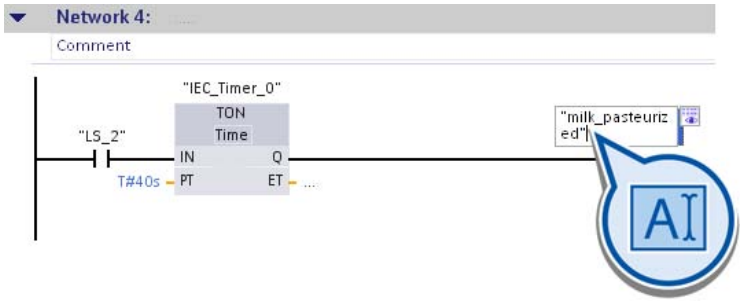


When the light barrier in the heating chamber detects a bottle, the signal state of the tag "LS_2" changes from "0" to "1". The period that is specified at the input PT of the time instruction will be started.

7. Insert the "Output coil" instruction at the end of the rung.



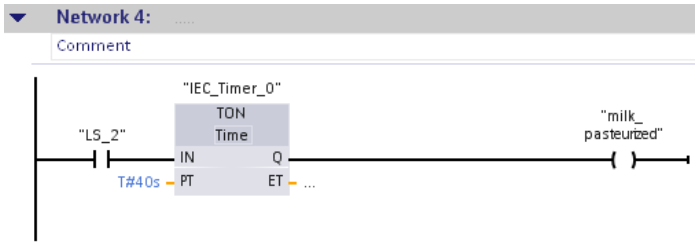
8. Interconnect the instruction "Output coil" with the "milk_pasteurized" tag.



The "milk_pasteurized" tag is set to the signal state "1" after expiration of the period specified at the input PT.

Result

You have programmed the duration of the pasteurization process.



When a milk bottle activates the light barrier "LS_2", the heating chamber is switched on and the pasteurizing time will be started. After the end of the period the "milk_pasteurized" tag is set to the signal state "1". The conveyor moves and the milk bottle is transported to the end of the conveyor.

In the next section you will program a status light that indicates the operating mode of the heating chamber.

3.2.6 Programming the status light

Introduction

The following steps show you how to program the status light. The status light indicates the operating modes of the heating chamber. When the heating chamber is in operation the tag LED is set to the signal state "1" and the display on the HMI device is switched on. When the heating chamber is switched off the tag LED has the signal state "0" and the display on the HMI device is switched off. You will program the display on the HMI device in the next section.

You program the status light in the fifth network of the organization block "Main [OB1]" with the help of the following instructions:

- Normally open contact
- Output coil

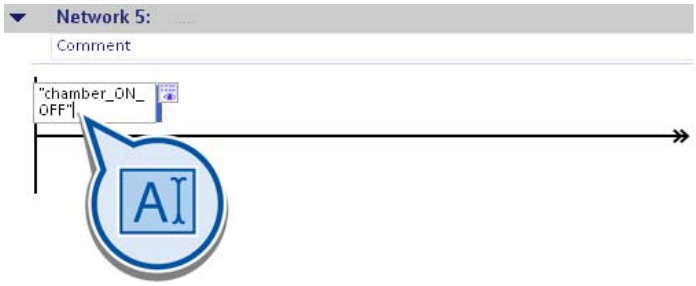
Requirements

- The organization block "Main [OB1]" is open.
- Networks 1 to 4 of the organization block "Main [OB1]" have been programmed.

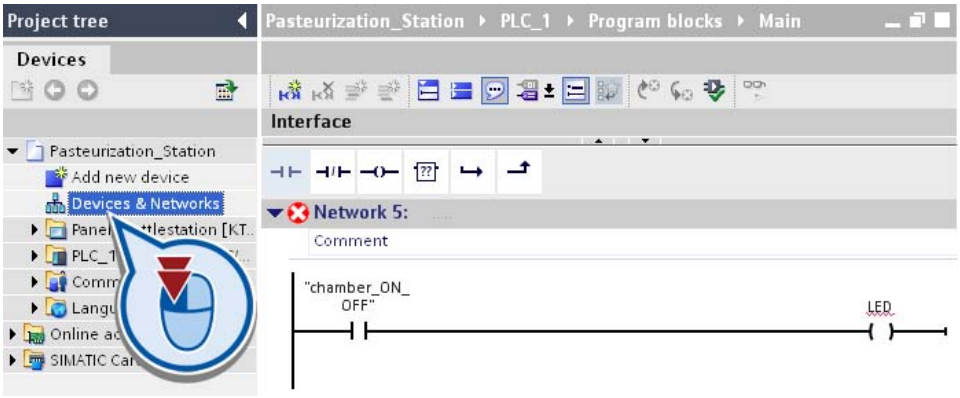
Procedure

To program the status light so that it indicates the operating mode of the heating chamber, follow these steps:

1. Insert a normally open contact into the fifth network of the "Main [OB1]" organization block.
2. Interconnect the normally open contact with the tag "chamber_ON_OFF".



3. Insert the "Output coil" instruction at the end of the rung.
4. Enter the name LED in the operand placeholder of the "Output coil" instruction and confirm the entry with the Enter key.
5. Open the device and network editor.

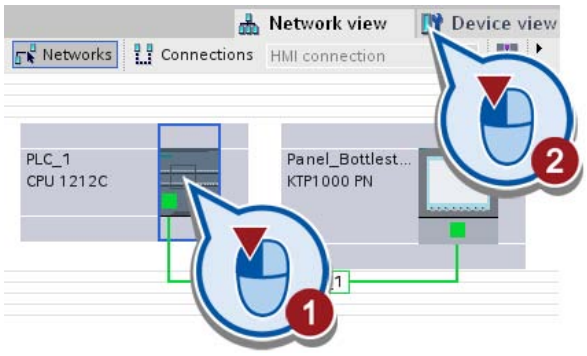


Note

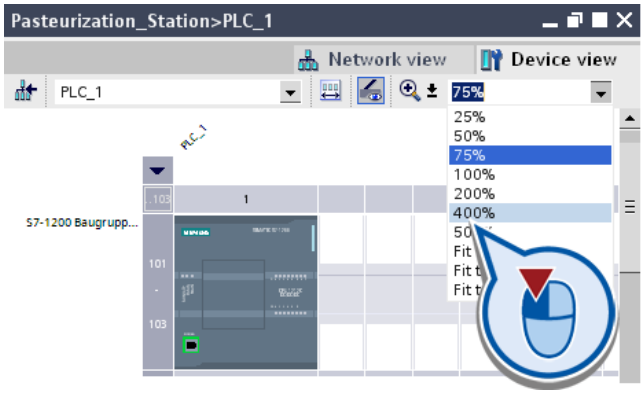
Network incomplete

A white cross in a red circle is displayed as icon in Network 5. The icon indicates that the interconnection or the tag definition in this network still has to be completed.

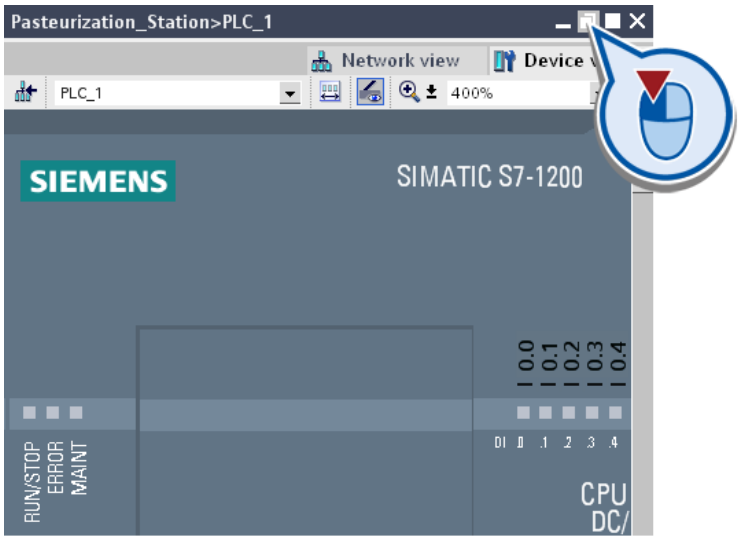
6. Switch to device view.



7. Zoom in the device view until the inputs and outputs of the PLC are easily visible.

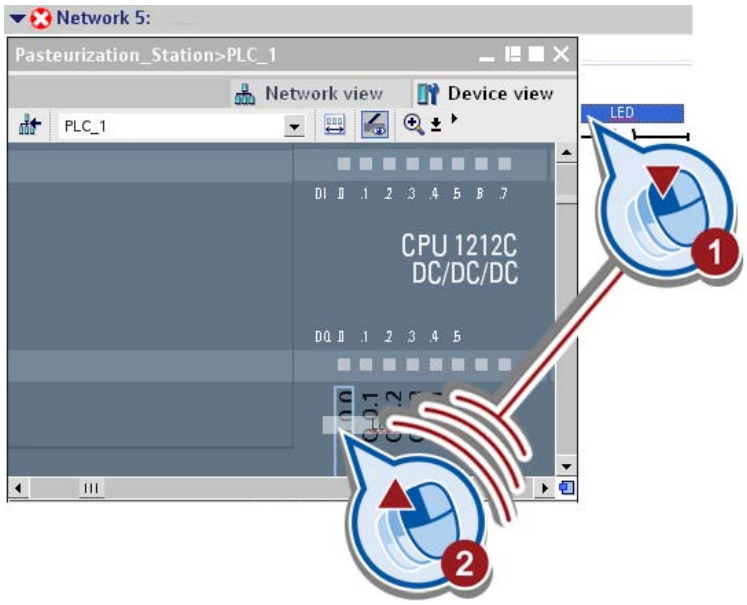


8. Detach the window of the device and network editor and position it beside the program editor.



9. Switch to the fifth network of the "Main [OB1]" organization block.

10. Interconnect the "LED" tag with the "Q 0.0" output of the PLC.



The "LED" tag is defined and displayed at the "Q 0.0" output of the PLC. The tags previously defined in the PLC tag table are displayed at the inputs of the PLC.

Note

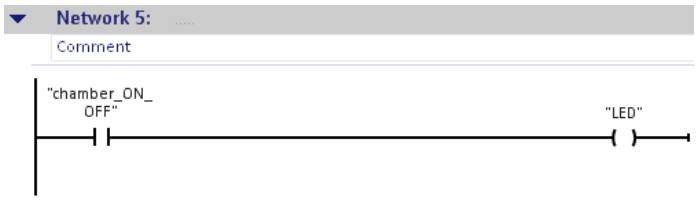
Defining tags

It makes no difference whether you define the tags at the inputs and outputs in the device view, the PLC tag table or the networks. The tags you have already defined will also be displayed in the device view.

11. Click the "Save project" button on the toolbar to save the project.

Result

You have programmed a status light to indicate the operating modes of the heating chamber.



When the heating chamber is switched on, the "chamber_ON_OFF" tag has the signal state "1". The "LED" tag is set to the signal state "1" by means of the "Output coil" instruction. As long as the heating chamber is switched on, the "LED" tag has the signal state "1". When the heating chamber is switched off, the value of the tags "chamber_ON_OFF" and "LED" changes to "0"

In the following sections you will test the created program with program status.

3.3 Test expanded program with program status

Introduction

The following steps show you how to test the created program with program status.

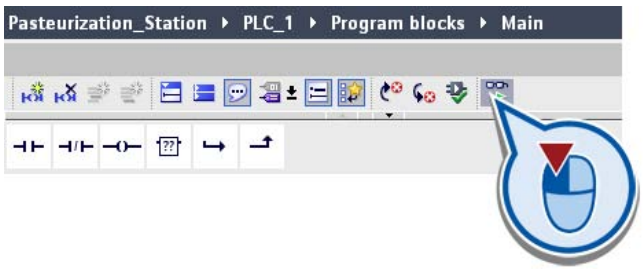
Requirements

- The PLC is configured.
- There is no voltage at the inputs and outputs of the PLC, because the modified values are overwritten by the module in online mode.
- The organization block "Main [OB1]" is opened in the program editor.

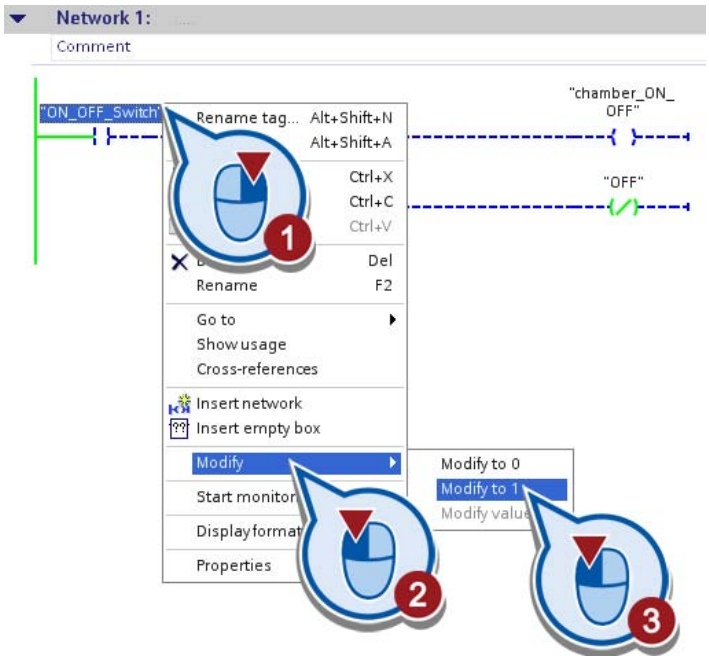
Procedure

To test the created program with program status, follow these steps:

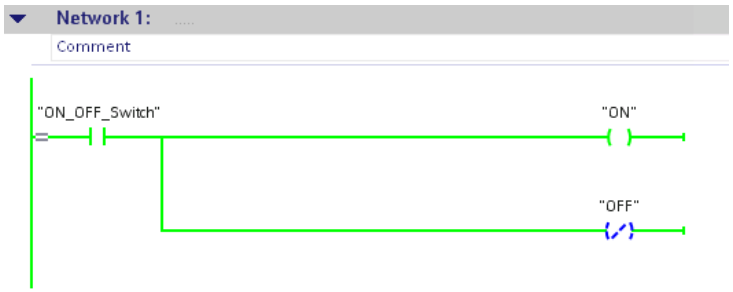
1. Load the program to the PLC and activate the online connection. For additional information see the "Loading the program to the target system (Page 48)" section.
2. Click the "Monitoring on/off" button on the toolbar of the program editor.



3. In Network 1, modify the "ON_OFF_Switch" tag to "1".



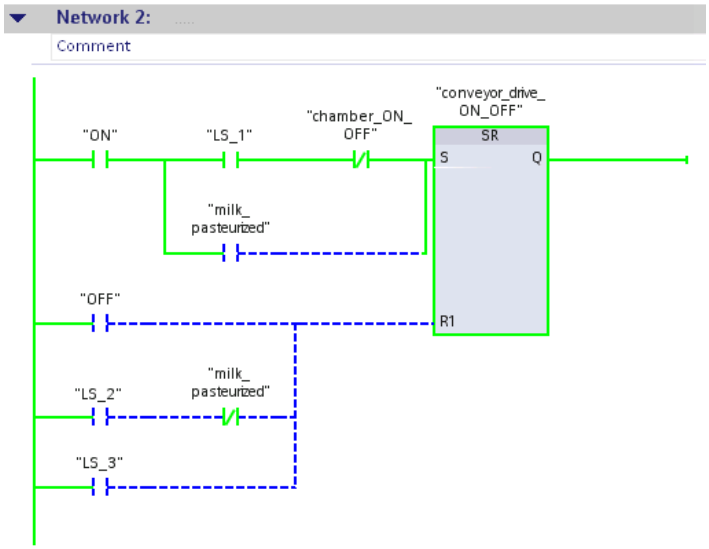
The "ON_OFF_Switch" tag is set to the signal state "1". The current flows through the normally open contact to the coils at the end of the network. The "ON" tag is set and the example machine thereby switched on. The OFF tag remains reset to "0" and has no further effect.



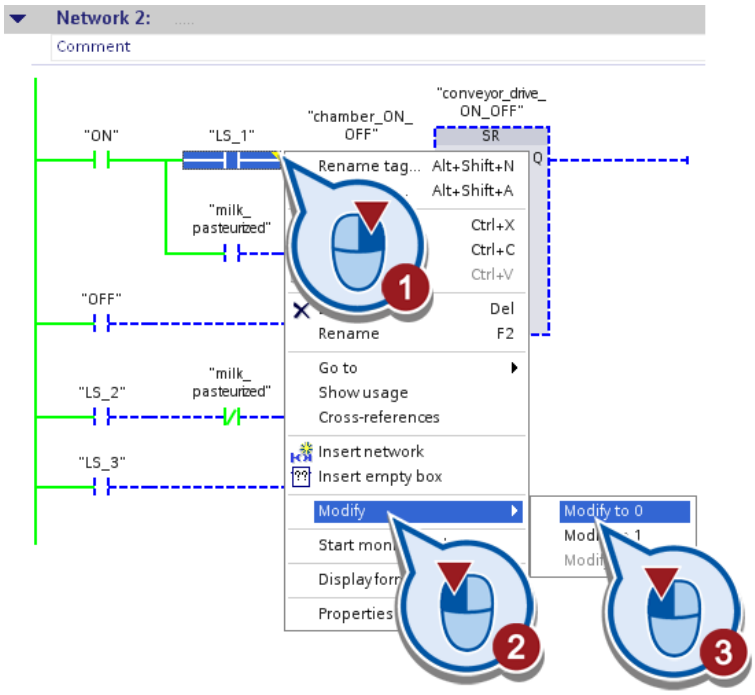
4. In Network 2, modify the "LS_1" tag to "1".

   The "LS_1" tag is set to the signal state "1". This simulates that a bottle is detected by the light barrier "LS1" at the start of the conveyor. The "chamber_ON_OFF" tag has the signal state "0", because the heating chamber is switched off during the simulation.

The power flows through the contacts of the main rung, resulting in the signal state "1" at the input S of the "Set/reset flip-flop" instruction. The "conveyor_drive_ON_OFF" tag is thereby set to the signal state "1" and the conveyor will be driven.



5. In Network 2, modify the "LS_1" tag to "0".



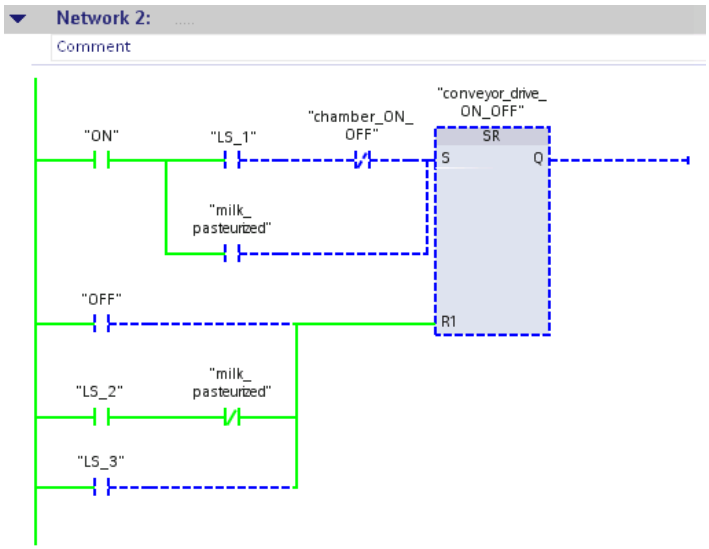
The LS_1 tag is reset to the signal state "0". This has the effect of simulating that the light barrier "LS1" is deactivated.

6. In Network 2, modify the "LS_2" tag to "1".

   In Network 2:

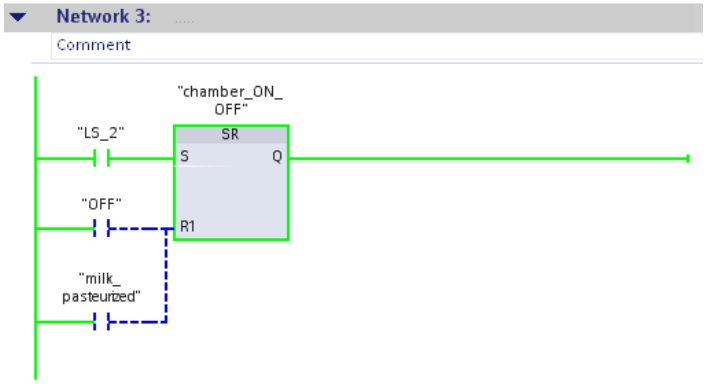
   The "LS_2" tag is set to the signal state "1". This setting simulates that the bottle was transported to the heating chamber and that the light barrier "LS2" was activated.

   The current flow is rerouted to the input R1 of the "Set/reset flip-flop" instruction. The conveyor_drive_ON_OFF tag is hereby reset and the conveyor stopped.



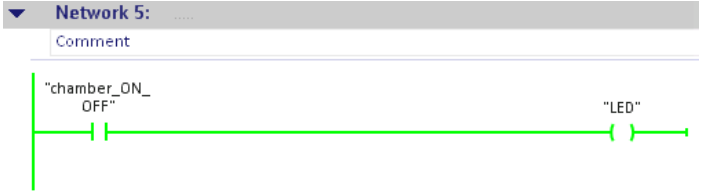
   In Network 3:

   The current flows through the contact of the main rung to the input S of the "Set/reset flip-flop" instruction. The chamber_ON_OFF tag is hereby set to the signal state "1", and the heating chamber is switched on.



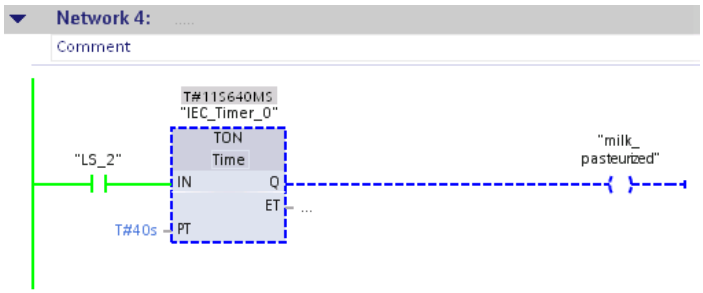
   In Network 5:

   As soon as the heating chamber is switched on (Network 3), the LED tag is set to the signal state "1" and the status light will be activated.

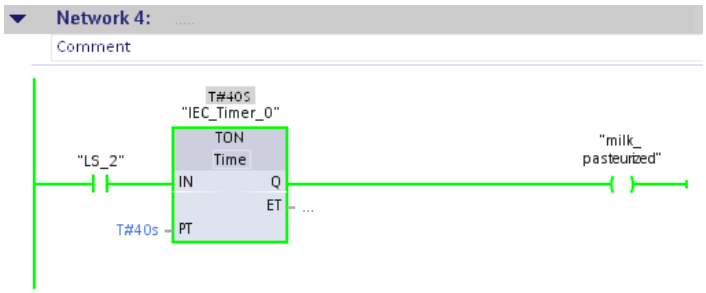


In Network 4:

A rising edge is detected at the IN input of the "On delay" time operation and the pasteurizing period has started.

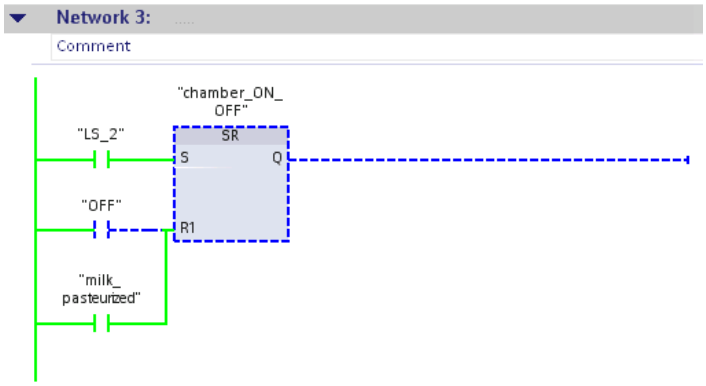


At the end of the period the milk is pasteurized and the "milk_pasteurized" tag will be set to the signal state "1".



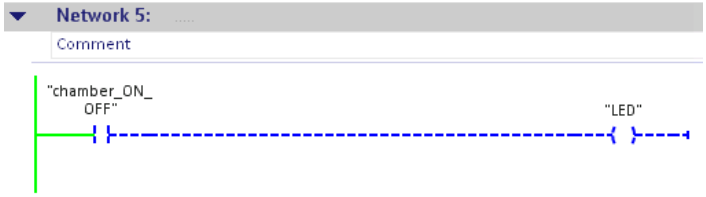
In Network 3:

The heating chamber is switched off when setting the "milk_pasteurized" tag.



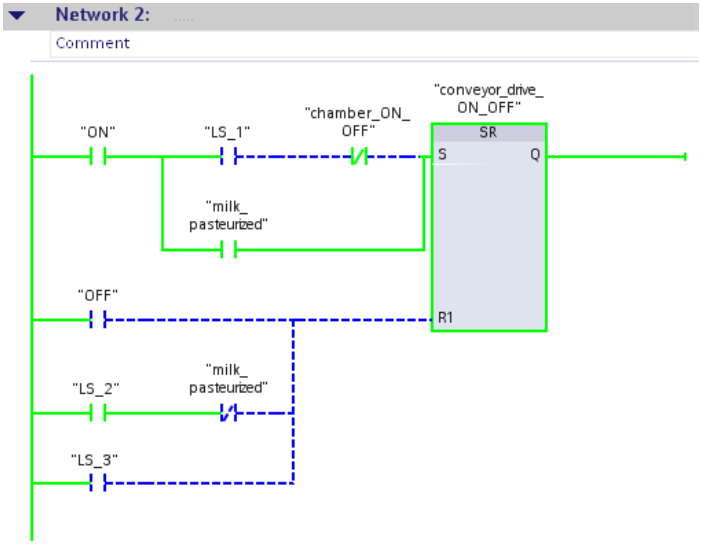
In Network 5:

When the heating chamber is switched off, the status light is deactivated.



In Network 2:

The conveyor is set into motion once again.



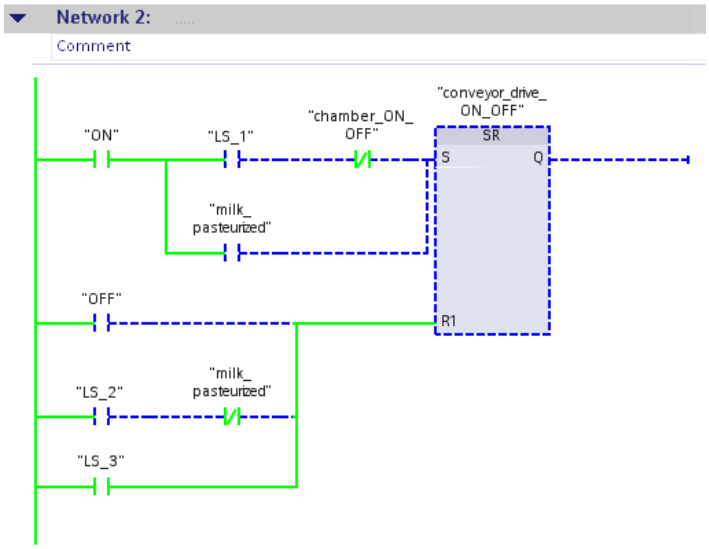
7. In Network 2, modify the "LS_2" tag to "0".

   The LS_2 tag is reset to the signal state "0". This state simulates that the bottle has left the heating chamber.

8. In Network 2, modify the "LS_3" tag to "1".

   The "LS_3" tag is set to the signal state "1". This state simulates that the bottle has been transported to the end of the conveyor and is detected by the light barrier "LS3"

   The conveyor_drive_ON_OFF tag is reset to the signal state "0" and the conveyor will stop.



9. Terminate the online connection.

Result

You have tested the program and checked the program execution.

In the next section you will add more graphic objects to the HMI screen. This allows you to visualize the flows of the expanded program.

3.4 Expand HMI screen

3.4.1 Graphic object "Heating chamber"

Introduction

The following steps show you how to add the graphic representation of a heating chamber to the HMI screen. Use simple static graphic objects to display the heating chamber.

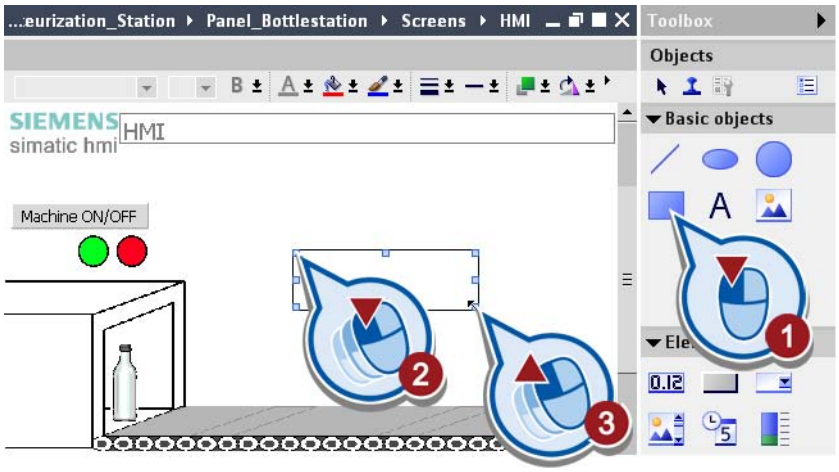
Requirements

- The HMI screen from the "Simple Example" chapter has already been created or the "Simple Example (Page 17)" project has already been loaded.
- The HMI screen created under "Panel_Bottlestation" is open.

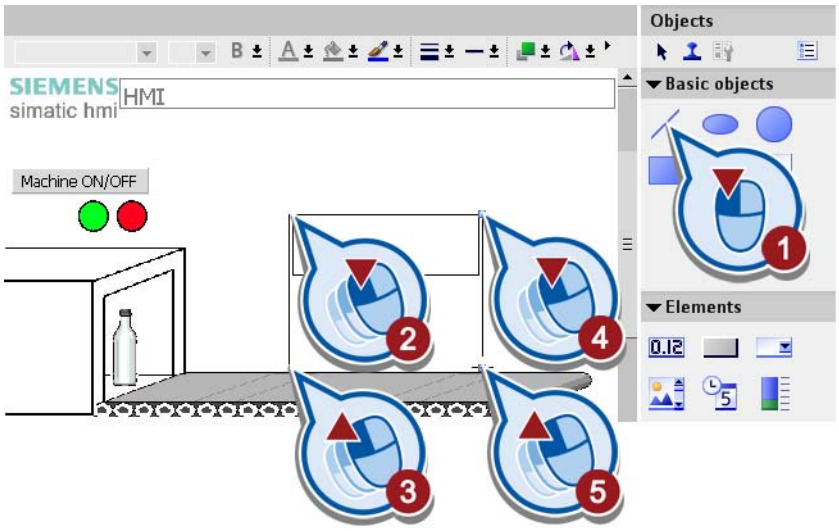
Procedure

To add the representation of the heating chamber to the HMI screen, follow these steps:

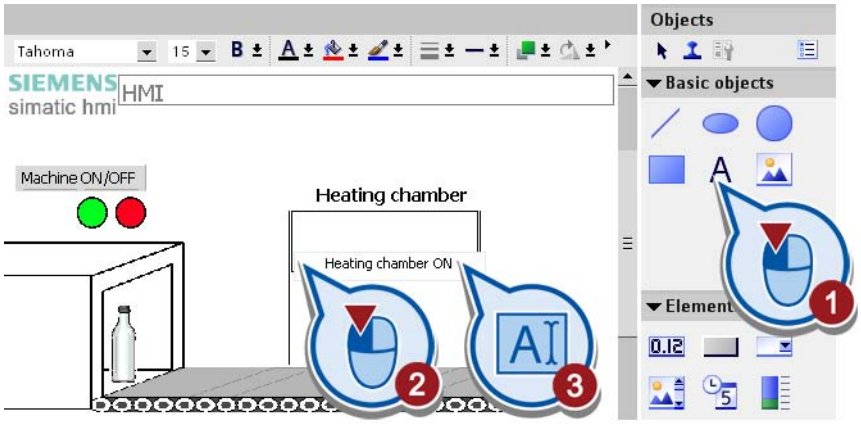
1. In the middle of the screen draw a rectangle above the conveyor.



2. Draw a vertical line on both the left and right side of the rectangle.



3. Create two text blocks, "Heating chamber" and "Heating chamber ON".



4. In the Inspector window, adjust the formatting of the text under "General" and "Appearance".

Result

You have added the static screen element of the heating chamber to the HMI screen.

3.4.2 Graphic object "Heating chamber LED"

Introduction

The following steps show you how to add the graphic representation of a LED to the HMI screen and animate it with the PLC tag "LED".

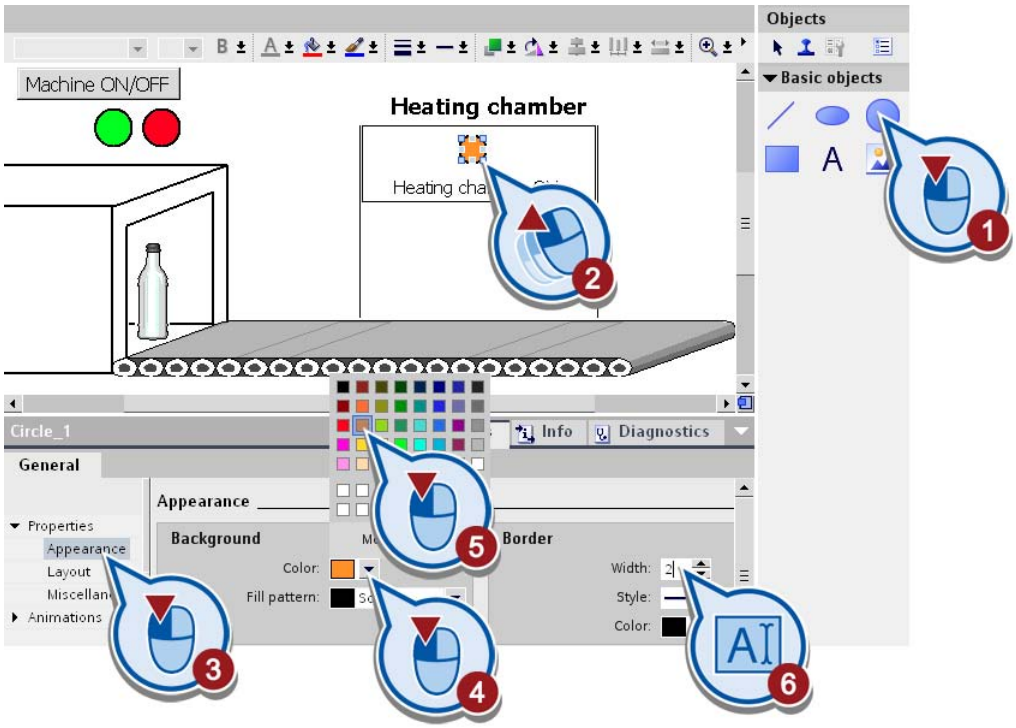
Requirements

- The program has been created.
- The HMI screen is open.

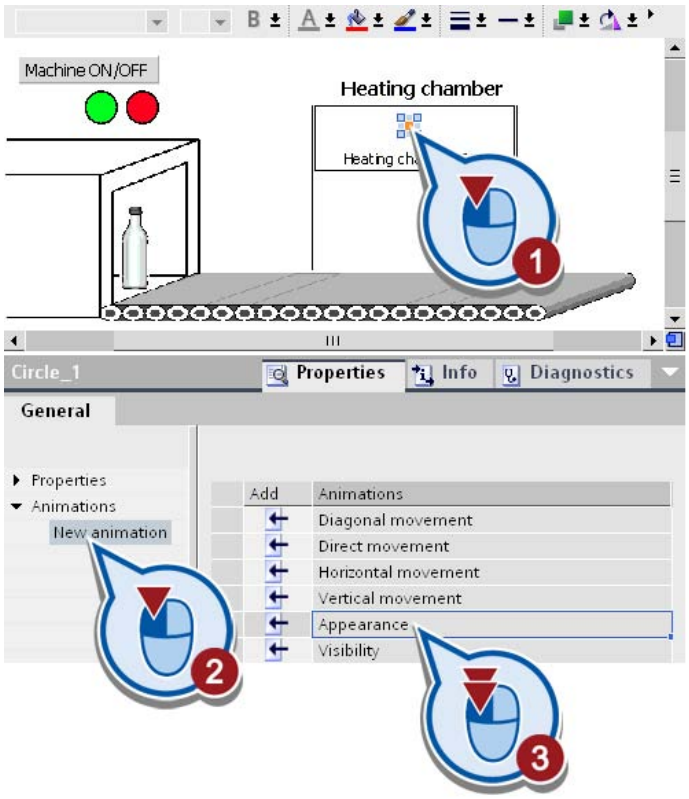
Procedure

To create the LED for the heating chamber and animate it, follow these steps:

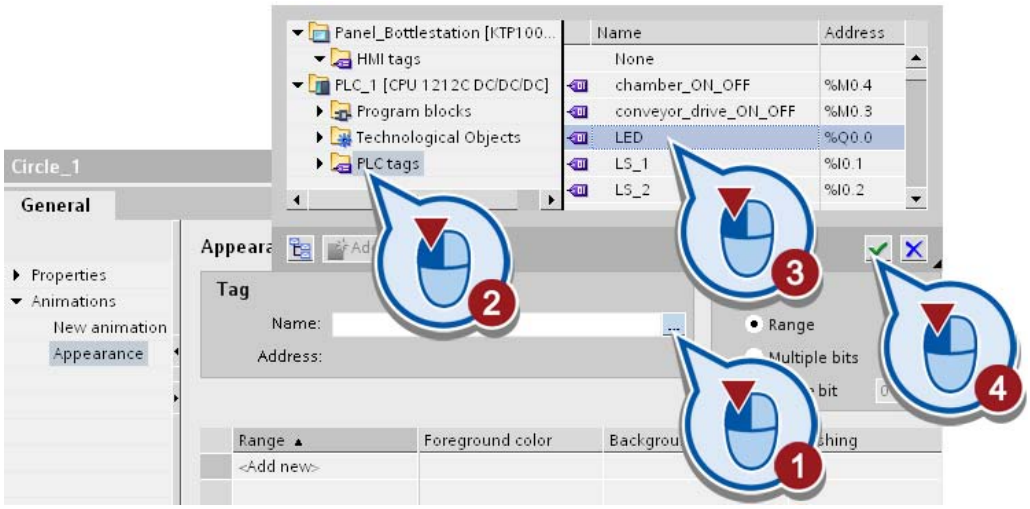
1. In the rectangle insert an orange circle, representing the heating chamber LED, in the center above the text block "Heating chamber ON".



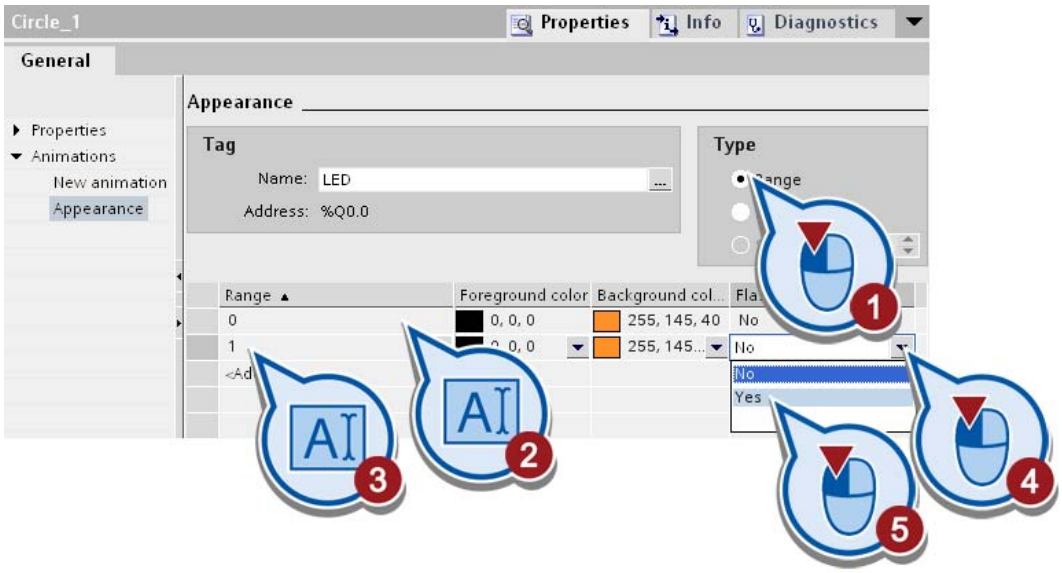
2. Create a new animation for the heating chamber LED.



3. Link the animation to the LED tag.



4. Activate "Flashing" for the value range "1" of the tag.



Result

You have added an LED to the HMI screen. If the "LED" tag takes the value "1", the activation of the heating chamber is displayed in the HMI screen by the flashing of the created LED.

In the next section you will add the graphic representation of the light barriers to the HMI screen.

3.4.3 Graphic objects "Light barriers"

Introduction

The following steps show you how to add the light barriers "LS1", "LS2" and "LS3" to the HMI screen. You use the basic object "circle" to represent the light barriers. You link the light barriers you have created to the tags of the program and animate the representation.

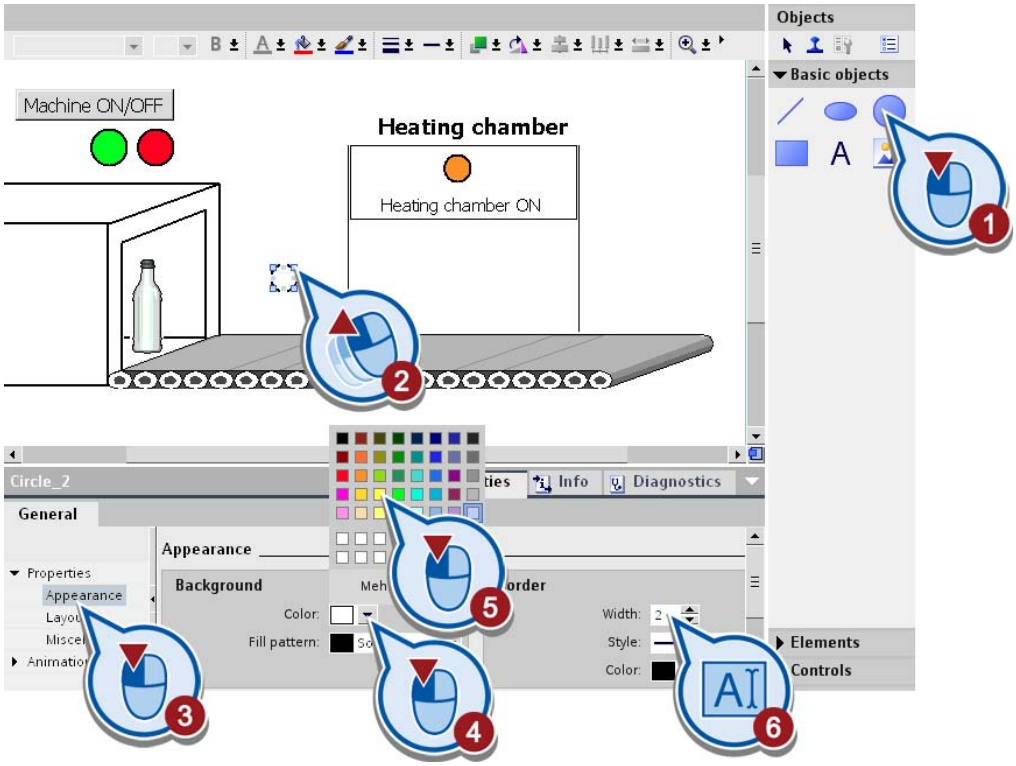
Requirements

- The control program has been created.
- The HMI screen is open.

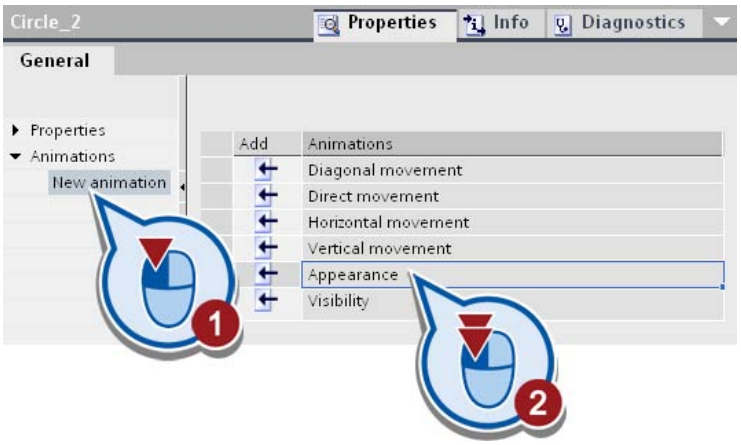
Procedure

To insert the representation of the light barriers in the HMI screen, follow these steps:

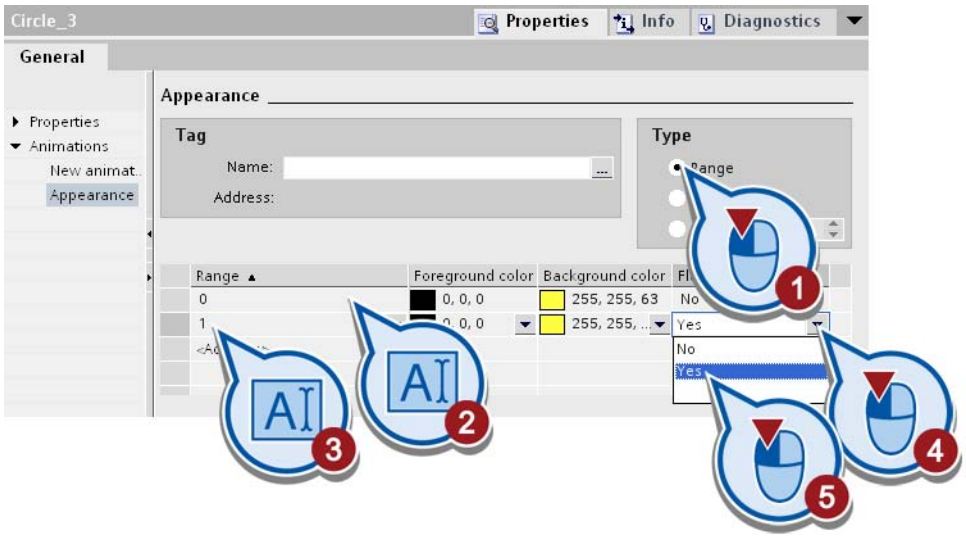
1. Insert a circle object to represent a light barrier with yellow background color and a border of "2".



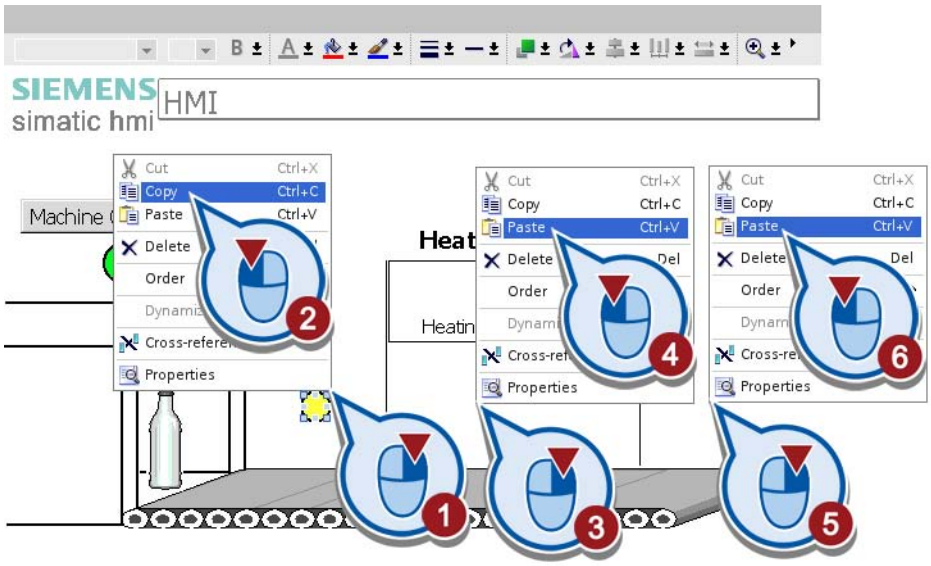
2. Create an "Appearance" animation for the light barrier.



3. Activate the "Flashing" function for the tag range "1", without previously defining a tag.

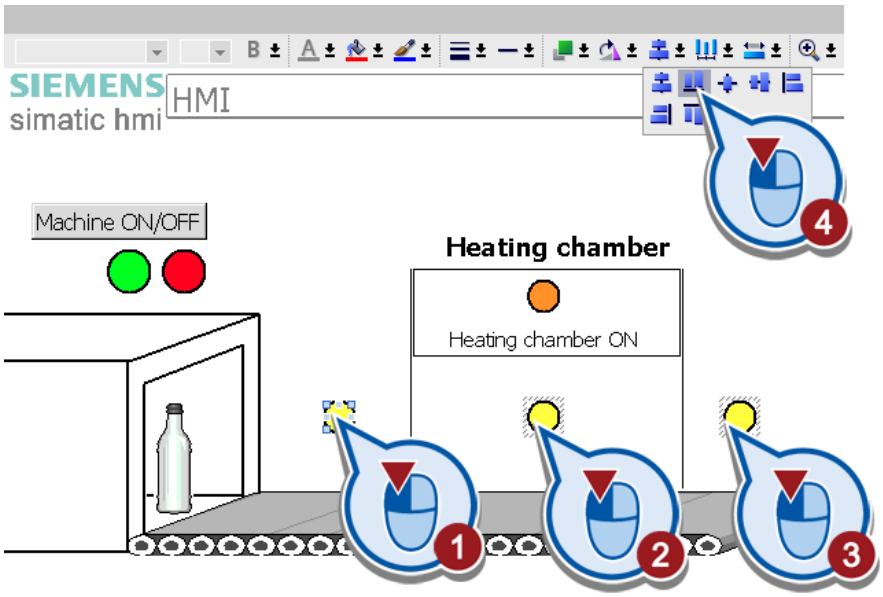


4. Use copy and paste to create two additional light barriers.

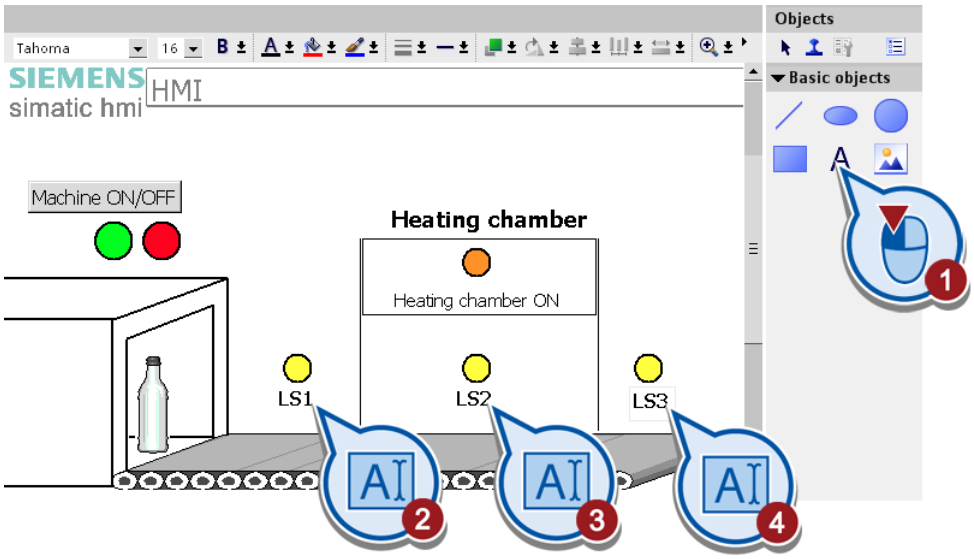


5. Position the light barrier in the of the middle heating chamber and at the end of the conveyor belt.

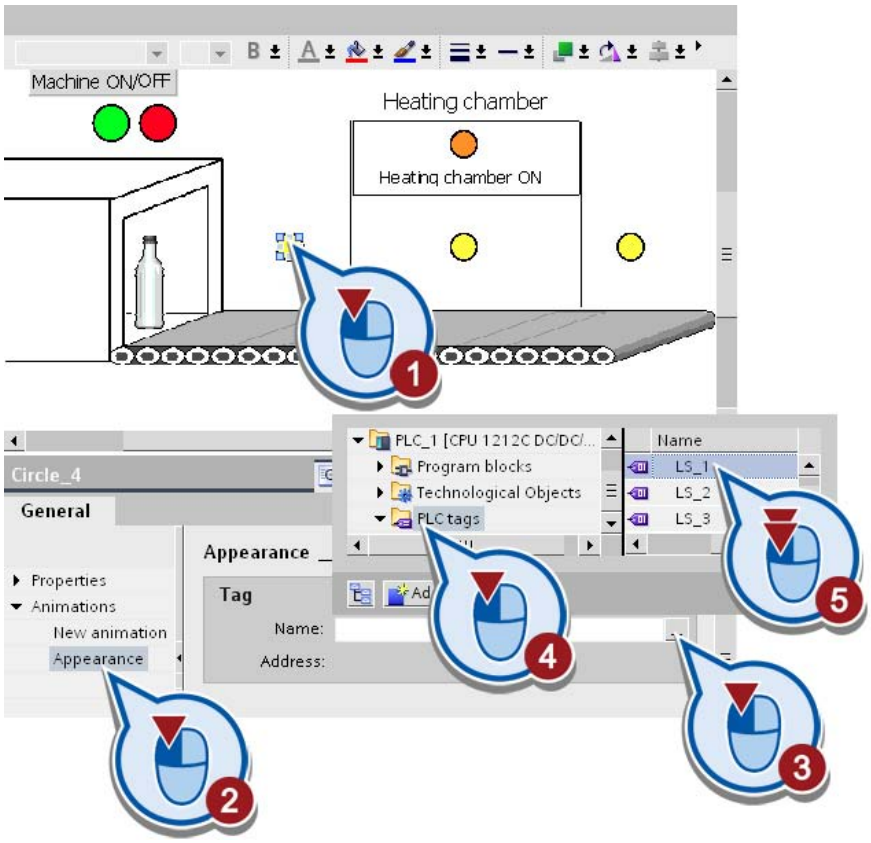
6. With the Shift key pressed, select all three light barriers and align them.



7. Label the light barriers "LS1", "LS2" and"LS3", respectively.



8. Link the appearance animation of the first light barrier to the PLC tag "LS_1".



9. Link the appearance animation of the second and third light barrier to the PLC tag "LS_2" and "LS_3".
10. Click the "Save project" button on the toolbar to save the project.

Result

You have added the light barriers "LS1", "LS2" and "LS3" to the HMI screen. You have linked these with tags and animated them.

In the next section you will simulate the screen to test the behavior of the dynamic objects.

3.5 Simulate HMI screen

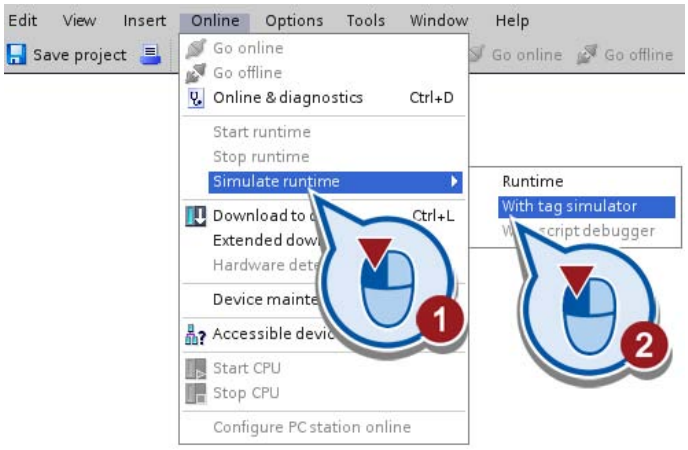
Introduction

The following steps show you how to test the created HMI screen with the Runtime Simulator. Use the Runtime Simulator to simulate the activation of the PLC input for the light barrier "LS1". For more information on the Runtime Simulator see the "Simulate runtime (Page 87)" section.

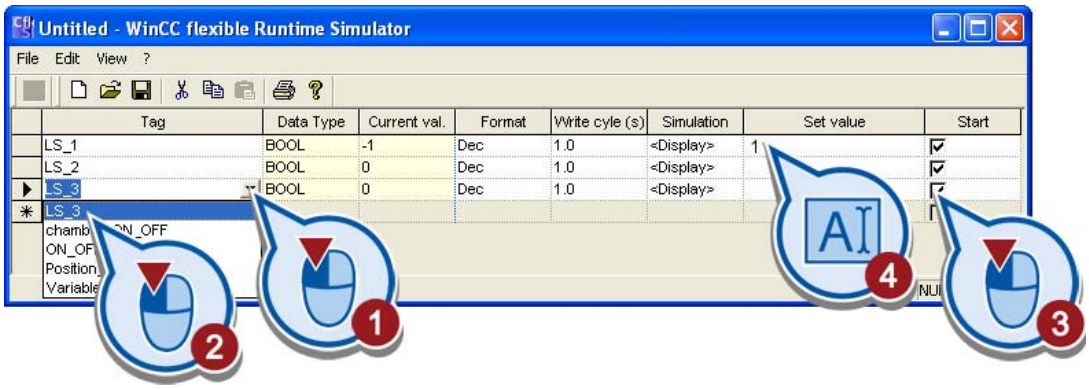
Procedure

To start the simulation of the created HMI screen, follow these steps:

1. Start the runtime simulation via the menu bar. The HMI device window must be active for this purpose. If the menu is inactive, first click on a free area within the HMI screen.



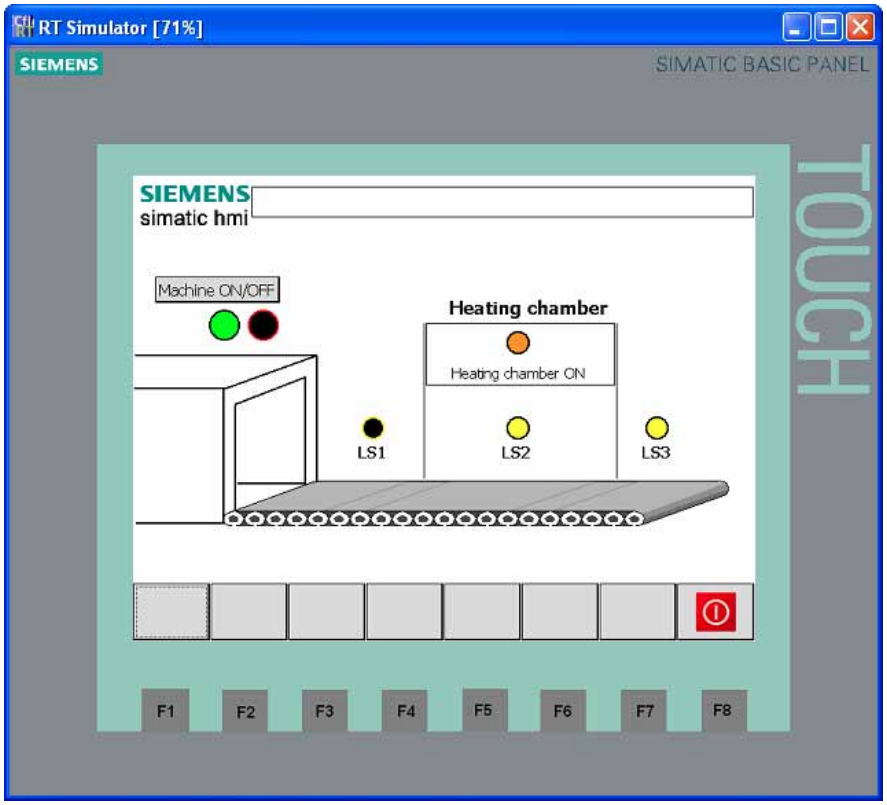
2. Set the value of the "LS_1" tag to "1". Click Enter to confirm the set value.



3. Switch to the "RT Simulator" window.

Result

The value for the PLC tag "LS_1" is simulated. The light barrier "LS1" flashes.



You can simulate values for all used PLC tags by assigning values in the "Set value" column. All PLC tags are Boolean values, which means they can only take the value "0" or "1".

Note

Tags in the Runtime Simulator

The HMI tags are displayed in the tag selection of the Runtime Simulator. PLC tags that are not linked to either events or an animation of the HMI screen are not listed in the selection of the "Tag" table column.

Testing in the HMI device

You have the option to test the HMI screen on an HMI device. For additional information see the Load HMI screen to the HMI device (Page 85) section.

Example "PID control"

4

4.1 Introduction

Loading a project

If you have skipped the previous chapters, you can load the project status at the end of the last chapter (see "Loading a project (Page 17)"). The project status at the end of this chapter is stored in the "Extended_Example.ZIP" file.

Introduction

A controller is always required when a certain physical value, such as temperature, pressure or speed, has to have a specific value in the process and this value can change depending on external conditions that cannot be foreseen.

Definition PID controller

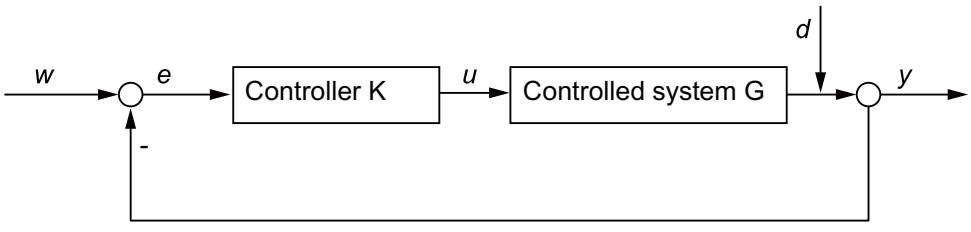
A PID controller consists of one proportional, one integral and one derivative element. It continuously detects the measured actual value of the controlled variable within a control loop and compares it with the desired setpoint. The PID uses the resulting control deviation to calculate a controller output that adjusts the controlled variable as quickly and smoothly as possible to the setpoint.

Definition control loop

A control loop is made up of a controlled system, the controller, the measuring element (sensor) and a final controlling element.

- In this example the PID control SIMATIC S7-1200 is used as the controller.
- The measuring element in this example is a sensor that measures the temperature inside the heating chamber.
- The final controlling element is the heater that is directly actuated by the PLC.

The following wiring diagram contains a typical control loop:



The setpoint "*w*" has been predefined. The setpoint in the following example is a temperature setting of 75°C in the heating chamber. The error (*e*) is calculated from the setpoint (*w*) and the actual value (*y*). The error will be converted by the controller (K) into a controlled variable (*u*). The controlled variable changes the actual value (*y*) via the controlled system (G). The controlled system (G) in this example is the temperature control in the heating chamber, for example, by increasing or decreasing the energy supply.

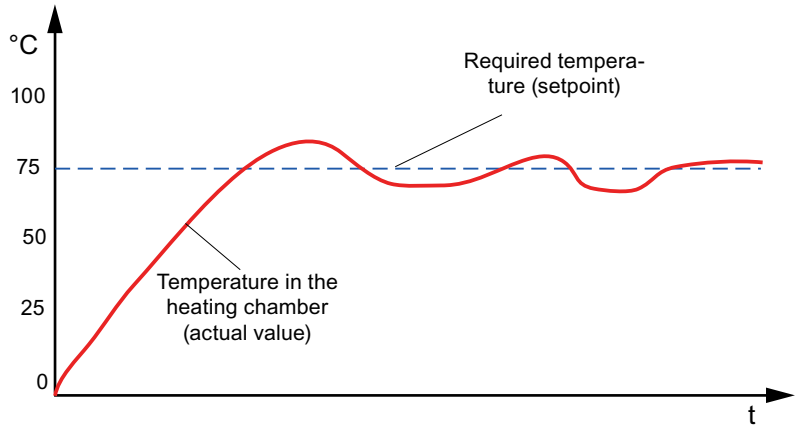
In addition to the controlled system (G), the actual value (*y*) can be altered through disturbance variables (*d*). A disturbance variable in this example may be an unwanted temperature change in the heating chamber, for example, caused by a change in the outside temperature.

Use of the PID controller

In the example project you use the PID controller to reach the desired temperature of 75°C as quickly as possible and to maintain the setpoint as constant as possible.

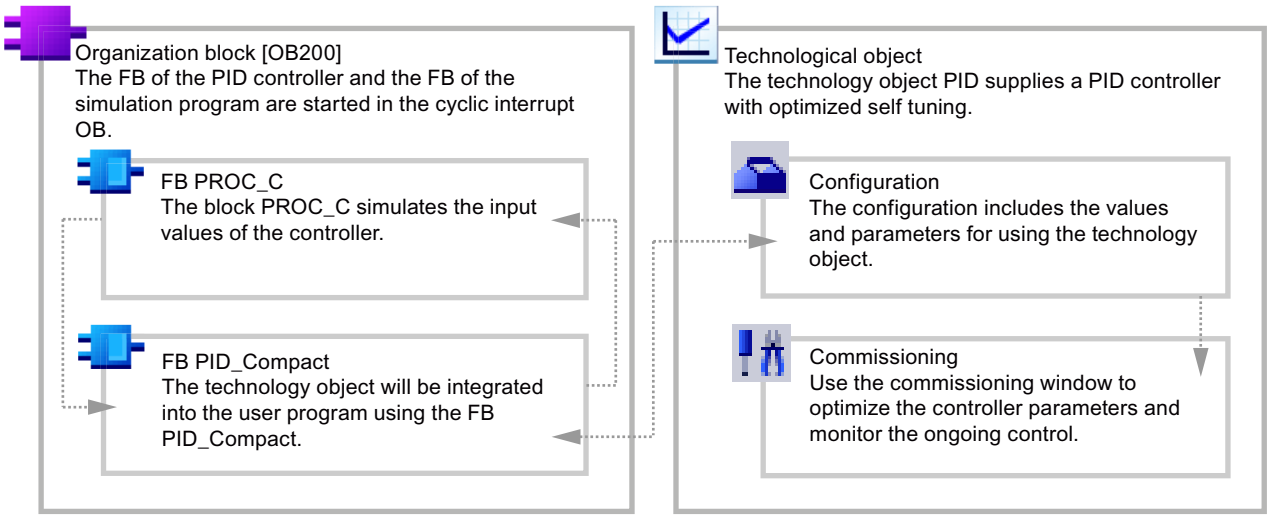
In this example, the setpoint can be exceeded because the heating element continues to emit heat even after it has been switched off. This effect is known as "overshoot"; it occurs when there is a time delay between the control and the measurement of the actual value.

The following figure shows the possible characteristic of the temperature curve after the initial switching on of the machine:



Steps

The following figure shows an overview of all objects you are going to create:


Organization block [OB200]
The FB of the PID controller and the FB of the simulation program are started in the cyclic interrupt OB.
FB PROC_C
The block PROC_C simulates the input values of the controller.
FB PID_Compact
The technology object will be integrated into the user program using the FB PID_Compact.
Technological object
The technology object PID supplies a PID controller with optimized self tuning.
Configuration
The configuration includes the values and parameters for using the technology object.
Commissioning
Use the commissioning window to optimize the controller parameters and monitor the ongoing control.

Proceed as follows to create the objects:

- You create a second organization block [OB200] in which you will call up the blocks of the PID controller.
- You create the technological object "PID_Compact".
- You load the simulation block "PROC_C" to the organization block [OB200]. By using the simulation block, you require no hardware other than the PLC.
- You configure the technological object "PID_Compact".
  - Select the controller type.
  - Enter a setpoint for the controller.
  - Interconnect the actual value and the controlled variable of the technological object "PID_Compact" with the simulation block "PROC_C".
- You load the user program and carry out a controller optimization in the commissioning window of the technological window.

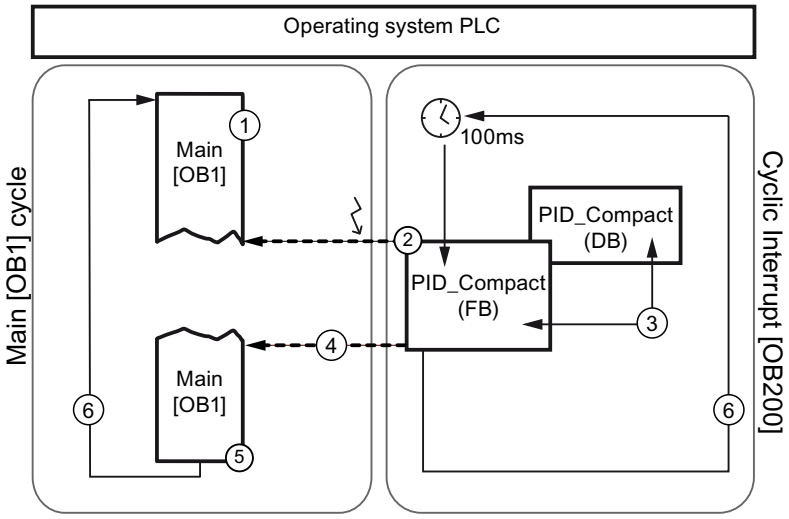
4.2 Set organization block for PID controller

Introduction

The blocks for the PID controller are created in a new organization block. A cyclic interrupt organization block, which you will now create, is used as new organization block.

Cyclic interrupt organization blocks serve to start programs in periodic time intervals independent of the cyclic program execution. Cyclic program execution will be interrupted by the cyclic interrupt OB and resume after the interrupt.

The following figure shows the program execution with a cyclic interrupt OB:



① Program starts with Main [OB1].

② A cyclic interrupt is triggered every 100 ms, which interrupts the program at any point (for example, during Main [OB1]) and executes the program in the cyclic interrupt OB.

In this case the program consists of a function block PID_Compact.

③ PID_Compact is executed and the values are written to the data block PID_Compact (DB).

④ After execution of the cyclic interrupt OB, Main [OB1] will be continued at the point at which it was interrupted. The values remain intact.

⑤ Main [OB1] is completed.

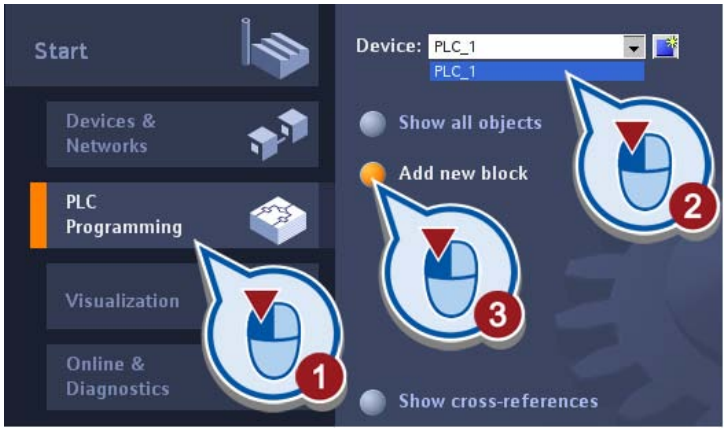
⑥ The program cycle starts again.

In the example project you use the cyclic interrupt OB to call the technological object "PID_Compact". The technological object "PID_Compact" is the image of the PID controller in the software. You can use this technological object to configure a PID controller, activate it and control its execution status.

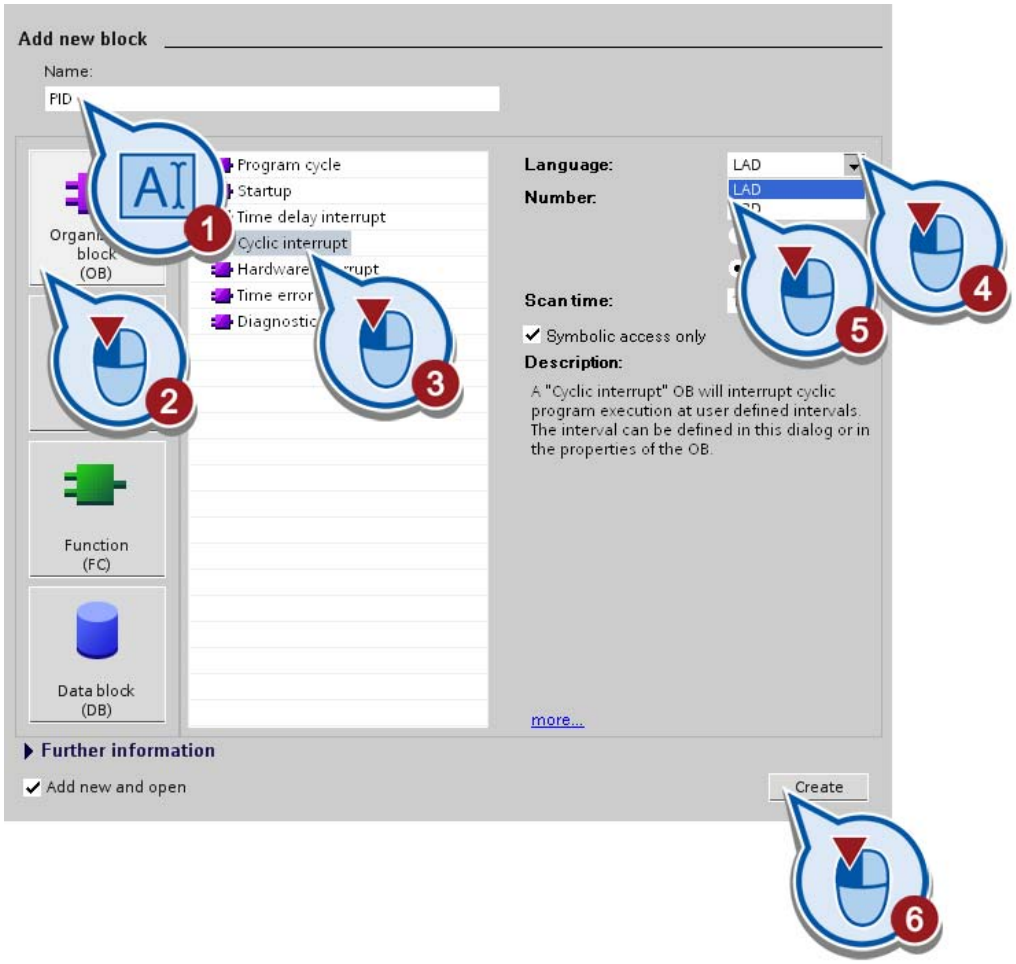
Procedure

To create a cyclic interrupt OB for the PID controller, follow these steps:

1. Open the portal view.
2. Add a new block to the existing PLC.



3. Create a cyclic interrupt OB with the name "PID". Make sure that the "Add new and open" check box is selected.



Result

The created cyclic interrupt OB is opened in the program editor in the project view. If the block does not open automatically, the "Add new and open" check box was not selected in the dialog. In this case, switch to the project view and open the program block in the project tree.

In the next section you will call the technological object "PID_Compact" in the created block.

4.3 Create technological object PID controller

Introduction

The following steps show you how to call the technological object "PID_Compact" in the cyclic interrupt OB "PID [OB200]".

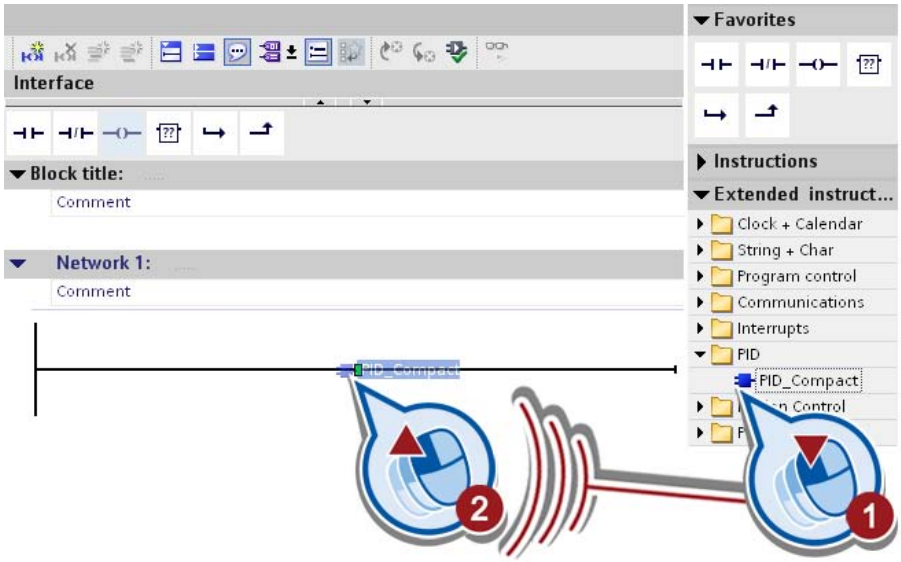
Requirements

- A project with a PLC S7-1200 has been created.
- A cyclic interrupt OB has already been created and opened in the project view.

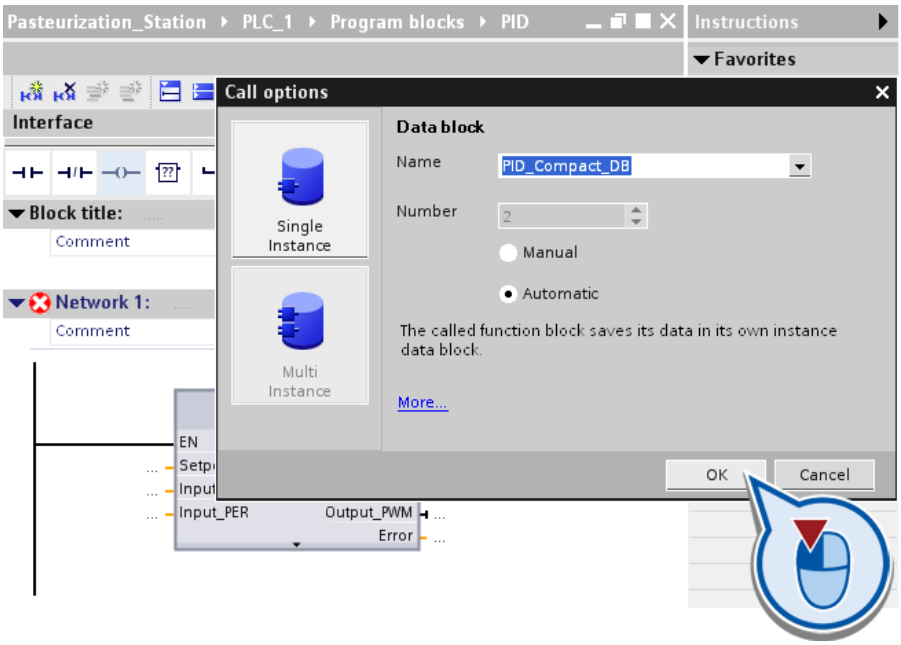
Procedure

To call the technological object "PID_Compact" in the cyclic interrupt OB "PID [OB200]", follow these steps:

1. Create the technological object "PID_Compact" in the first network of the organization block "PID [OB200]".



2. Confirm the creation of the data block for the technological object "PID_Compact".



Result

You have programmed the call of the technological object "PID_Compact" in the cyclic interrupt OB "PID [OB200]" and created the data block "PID_Compact_DB".

In the next section you will load a simulation block in the program to simulate the input and output values of the PID controller.

4.4 Load simulation block

Introduction

The following steps show you how to load the block "PROC_C" in the example project. The block simulates the input and output values of the PID controller. To use these, load a library in the example project and create the block in the second network.

Requirements

The organization block "PID [OB200]" is open in the project view.

Procedure

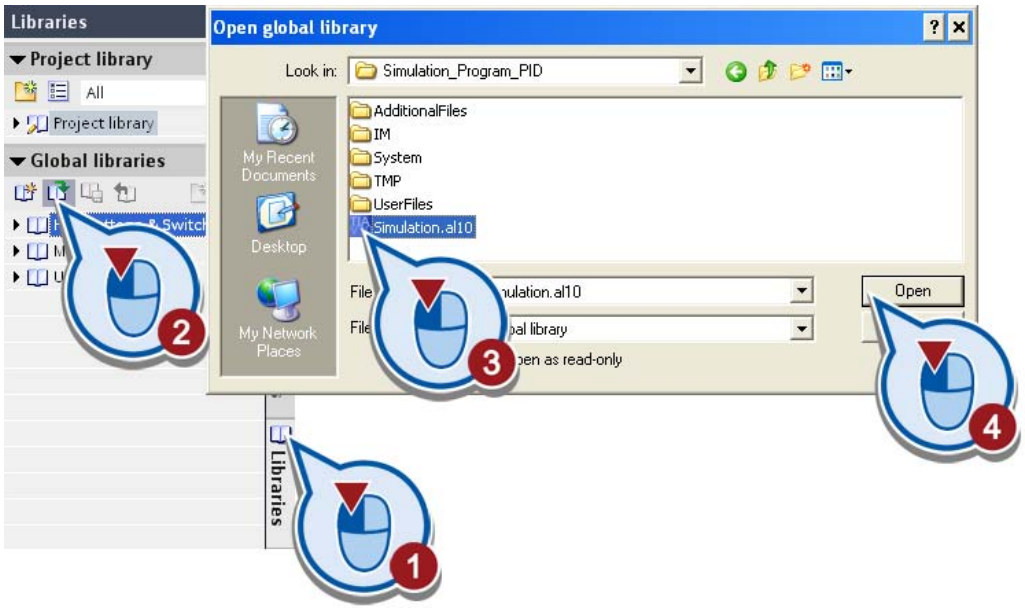
To open the library and copy the block, follow these steps:

1. Copy the file "Simulation Program PID.ZIP" from the following Internet address to your local hard disk and extract the file.

   http://support.automation.siemens.com/WW/view/en/40263542

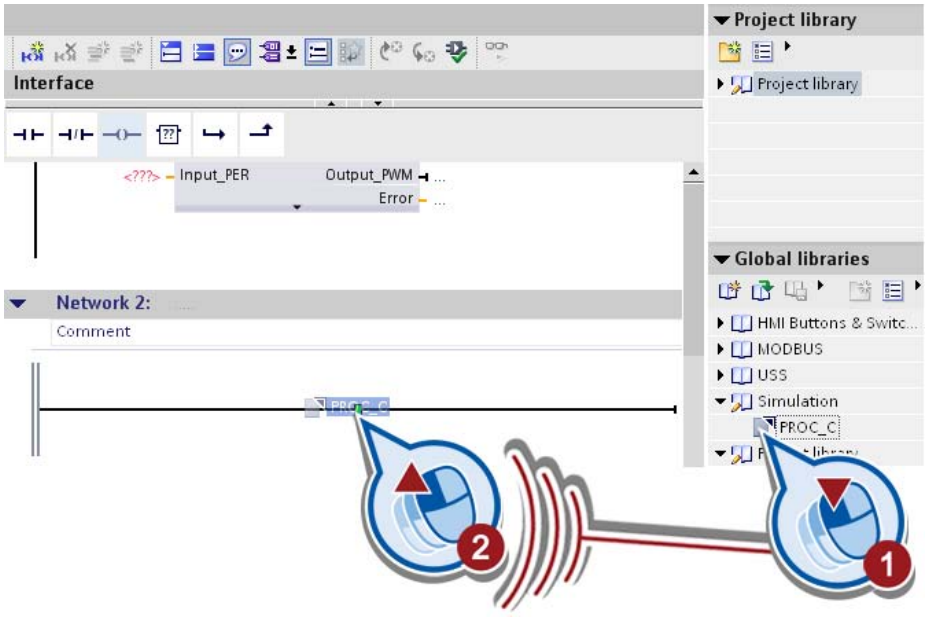
   Click on "Info" to see the ZIP files.

2. Extract the file "Simulation Program PID.ZIP".
3. Use the "Libraries" task card to open the global library "Simulation" from the directory of the extracted file.

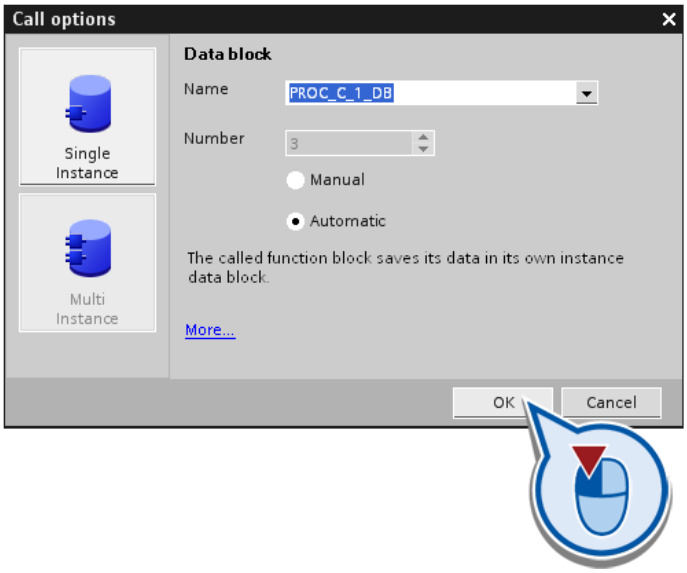


The library is loaded.

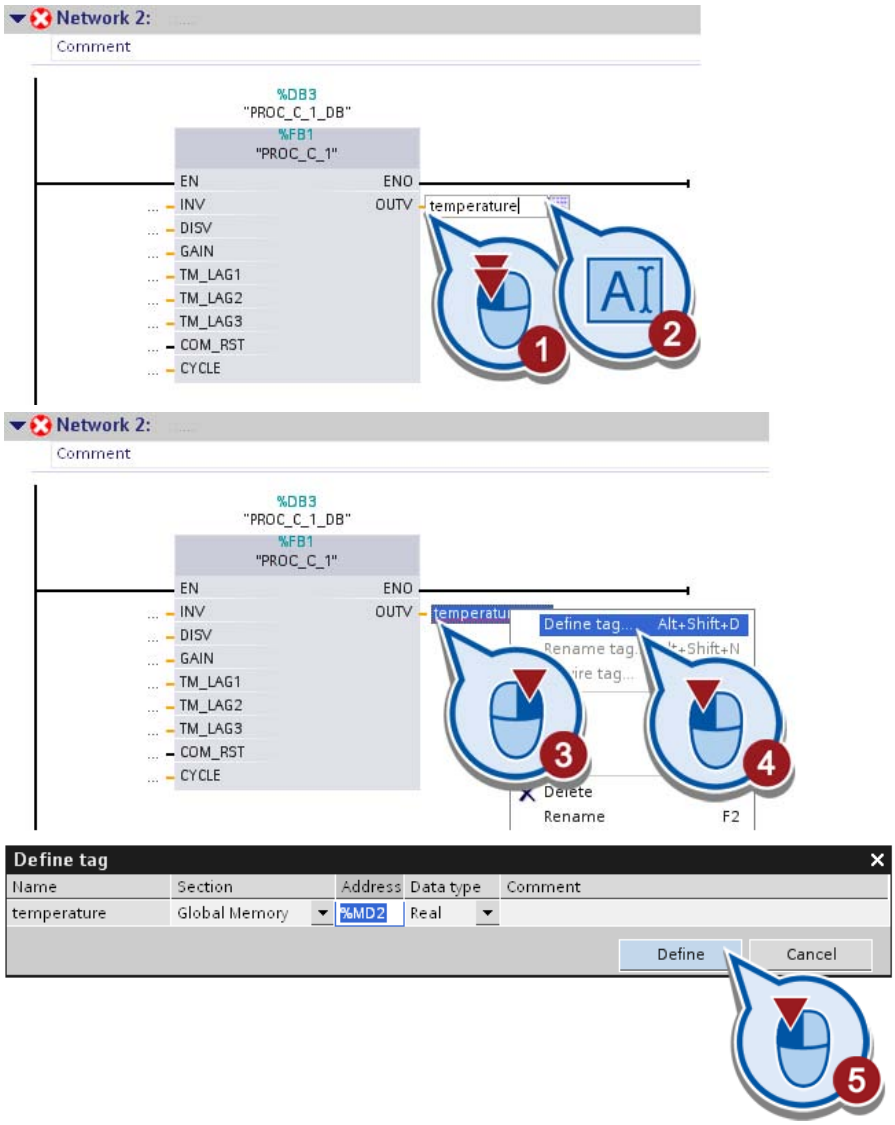
4. Copy the simulation block "PROC_C" to the second network of the organization block "PID [OB200]".



5. Confirm the creation of the data block for the simulation block "PROC_C".



6. Define a "temperature" tag on the OUTV parameter.

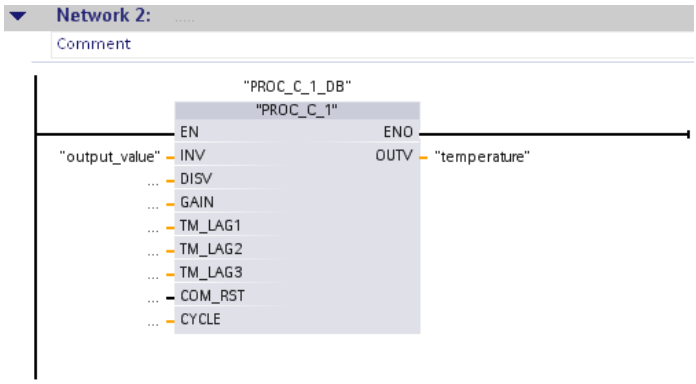


The value of OUTV parameter is stored in the "temperature" tag. The value of this parameter is the temperature value that is simulated by the "PROC_C" block.

7. In this same way, define a tag "output_value" on the INV parameter.

Result

You have loaded in the example project the block for the simulation of the input and output values of the PID controller.



When the cyclic interrupt OB "PID (OB200)" is executed, the "PROC_C" block simulates input and output values and loads these in the instance data block "PROC_C_DB". The values of the parameters INV and OUTV are mapped in the tags when the program is executed.

In the next section you will use the technological object "PID_Compact" to configure the PID controller and link its inputs and outputs to the corresponding values of the simulation block.

4.5 Configure PID controller

Introduction

The following steps show you how to use the technological object "PID_Compact" to configure the PID controller.

Settings for configuring the PID controller

- Controller type

  The controller type is used to define a pre-selection for the unit of the value to be controlled. In this example, "Temperature" with the unit "°C" is used as the controller type.

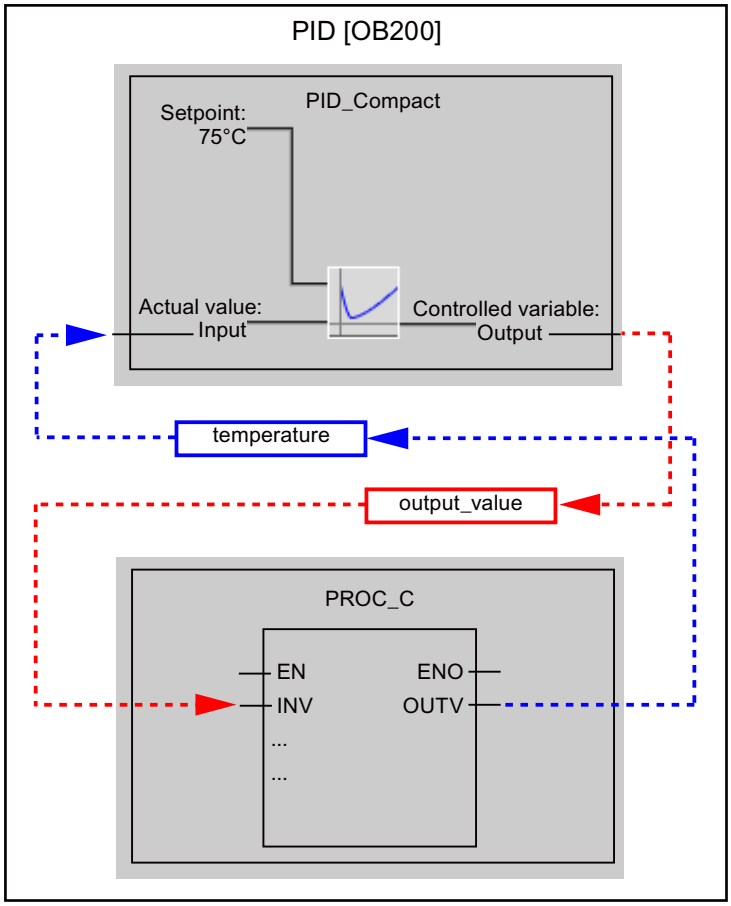
- Input/output parameters

  In this area you supply the input and output parameters for setpoint, actual value and the controlled variable of the technological object "PID_Compact". To use the PID controller without additional hardware, link the input and output parameters of "PID_Compact" to the "output_value" and "temperature" tags interconnected to the simulation block "PROC_C":

  – The actual value is simulated by "PROC_C" and used as input of "PID_Compact".

    In the example the actual value corresponds to the measured temperature in the heating chamber, which is mapped in the "temperature" tag.
  – The controlled variable by calculated by the technological object "PID_Compact" and is the output parameter of the block. The controlled variable is mapped in the "output_value" tag and is used as input value of "PROC_C".

The following figure shows how the technological object "PID_Compact" and the simulation block "PROC_C" are interconnected.



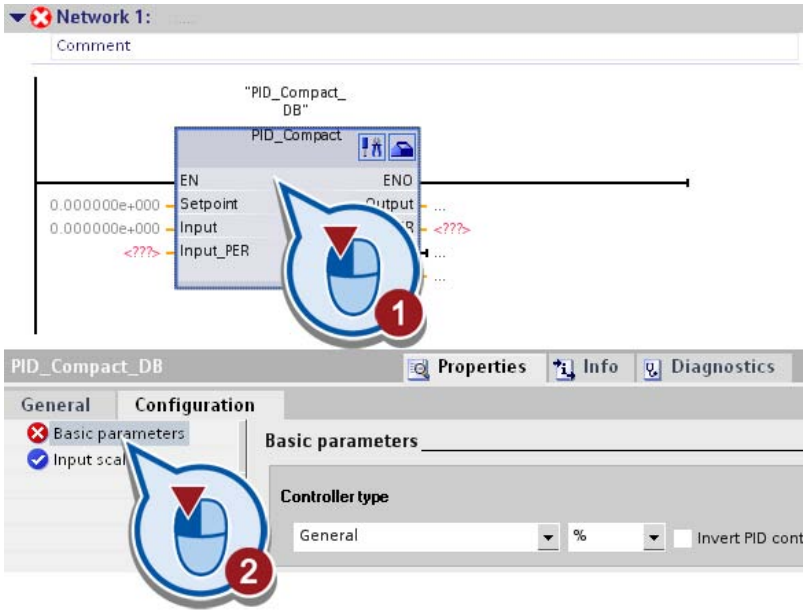
Requirements

- The cyclic interrupt OB "PID [OB200]" is open.
- The "PID_Compact" block is called in the "PID [OB200]" organization block.
- The "PROC_C" simulation block is called in the "PID [OB200]" organization block.

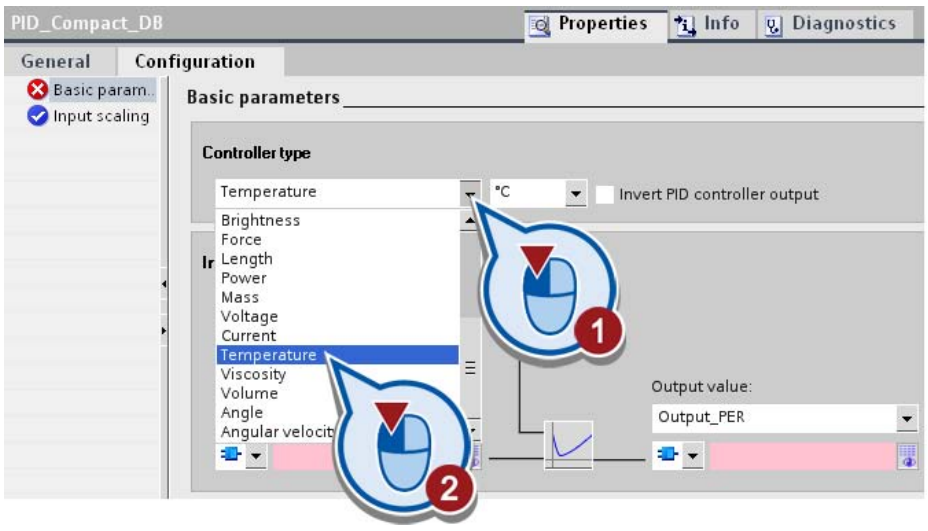
Procedure

To configure the technological object "PID_Compact" and interconnect it with the simulation block "PROC_C", follow these steps:

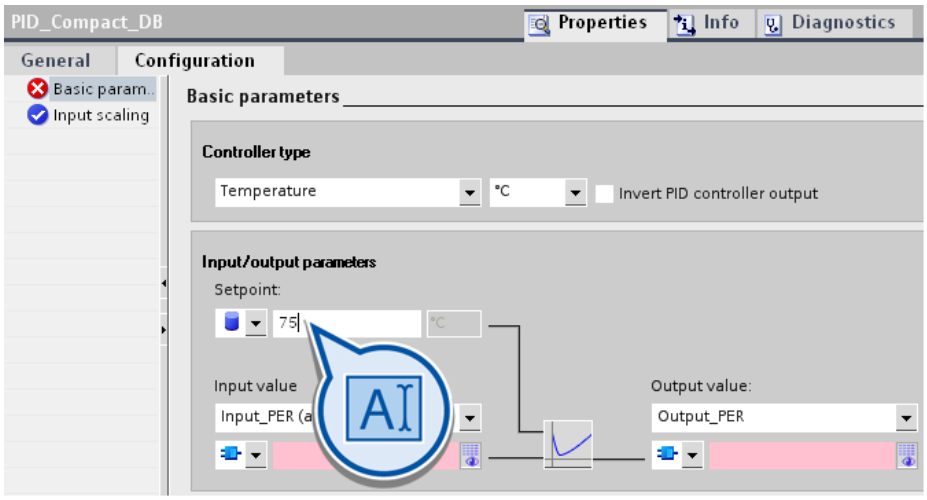
1. Open the configuration of the PID controller in the Inspector window.



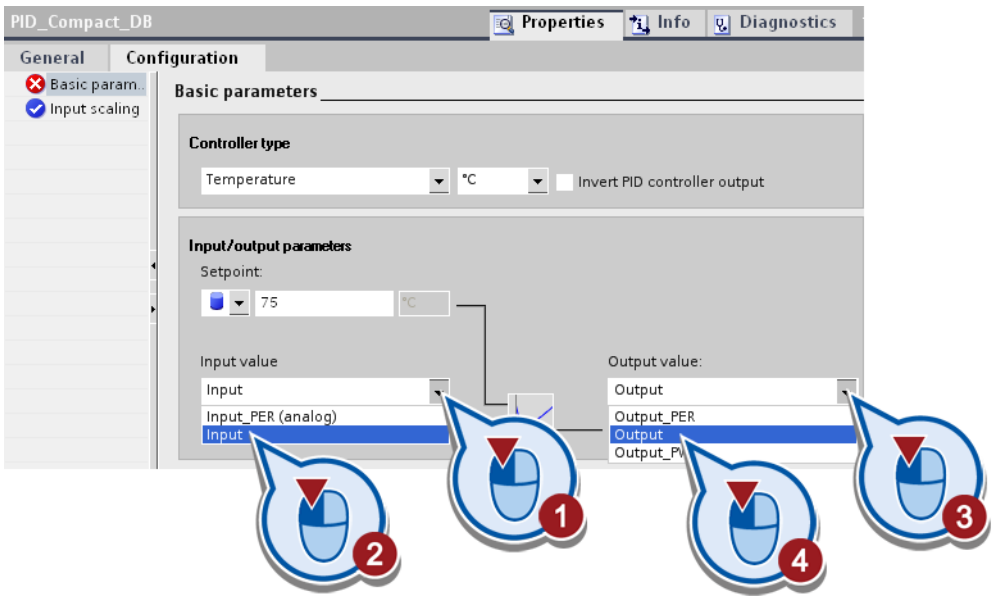
2. Select the controller type.



3. Enter the setpoint for the controller.



4. Select the "input" and "output" for the actual value and the controlled variable respectively. You hereby specify that the values from one tag of the user program are to be used.

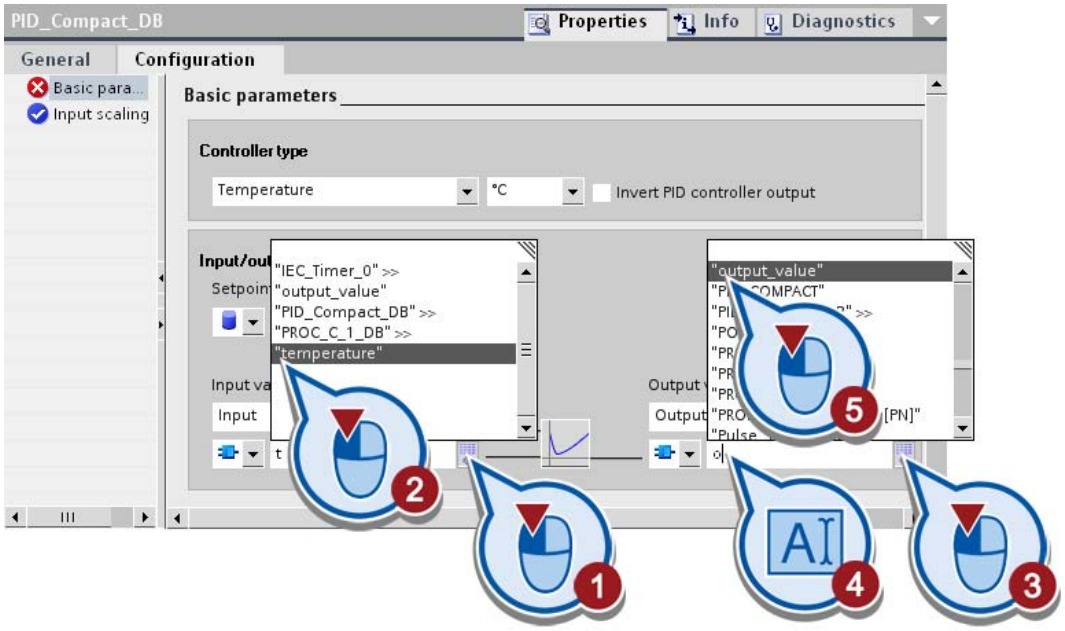


Note

Input(_PER) - Output(_PER)

Use input and output to supply the input or output parameters with the actual value of the user program. Use Input_PER and Output_PER to use an analog input as actual value or an analog output as manipulated value output. Use Output_PWM to use a digital switching output that is controlled via pulse width modulation. The manipulated variable is formed by variable on/off times in this case.

5. Interconnect the "temperature" tag with the actual value and the "output_value" tag with the controlled variable. When you enter the first letters of the tag the selection is filtered correspondingly by Intellisense®.



Note

Interconnecting blocks directly with parameters

The "temperature" tag will continue to be used in the following section. Otherwise you can also interconnect the "PID_ Compact" block directly with the parameters of the simulation block. Parameters are addressed as follows: "*Block name*".*Parameter*.

You can select from the drop-down list to the left of the input field of the actual value if the value will be the tag or parameter interconnected with the function block or the current value of the instance DB of the PID_Compact instruction.

Result

You have interconnected the PID controller (PID_Compact) and the simulation block "PROC_C". When the simulation is started, the PID controller receives a new actual value each time the organization block "PID [OB200]" is called.

4.6 Changing the heating chamber control

Introduction

The following steps will show you how to program the on/off control of the heating chamber depending on the machine operation. The heating chamber is turned on when the machine is started. The heating chamber temperature in this part of the sample project will not be constant and is controlled by the created PID controller.

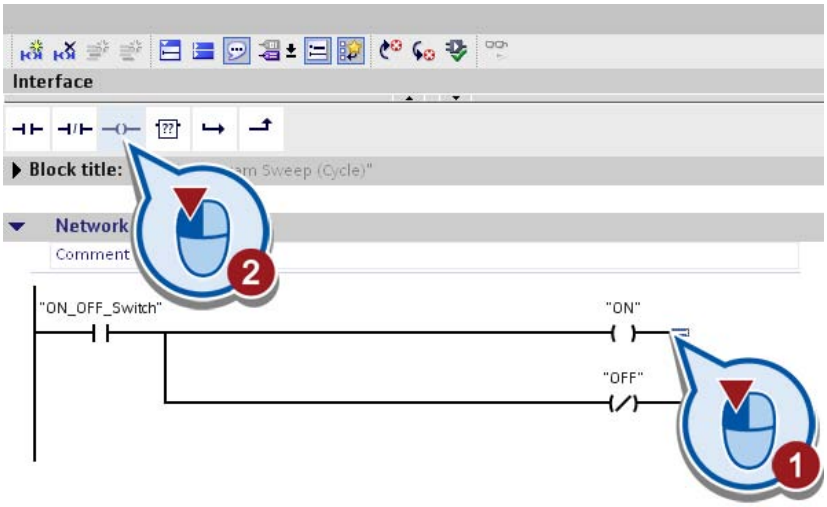
Requirements

- The organization block "Main [OB1]" is open.
- The networks of the organization block "Main [OB1]" have been programmed.

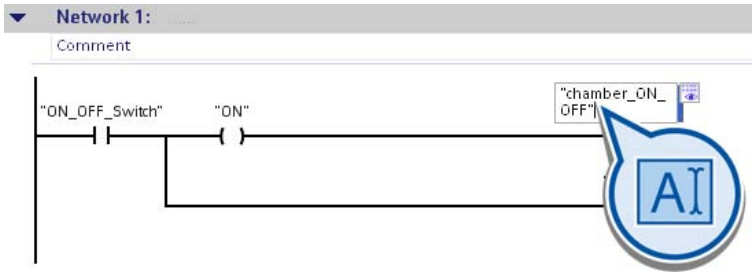
Procedure

To change the control for turning the heating chamber on or off, follow these steps:

1. Open the first network of the organization block "Main [OB1]".
2. Insert the "Output coil" instruction at the end of the main rung.

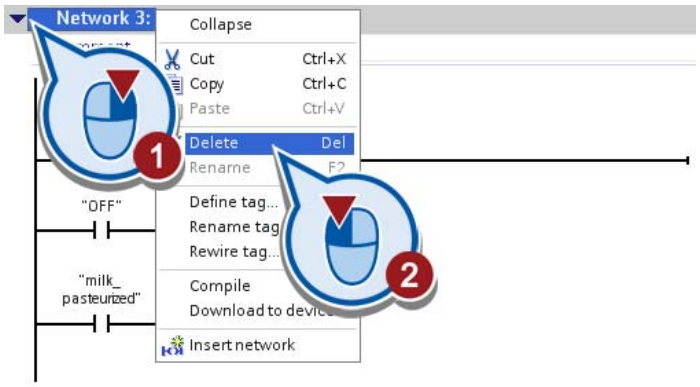


3. Interconnect the inserted instruction with the "chamber_ON_OFF" tag.

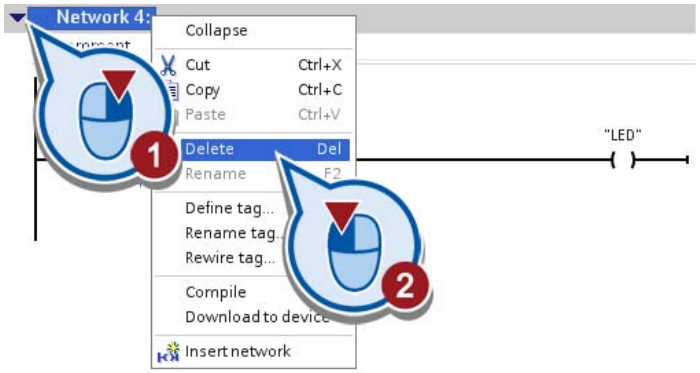


4. Delete the third network of the organization block "Main [OB1]".

   This program segment is no longer necessary because the on/off control of the heating chamber takes place via the program for controlling the machine in the first network of the organization block.



5. Delete the fourth network, in which the status lamp of the heating chamber is programmed. The status lamp is no longer necessary, because operation of the heating chamber takes place parallel to operation of the machine.



Result

You have programmed the on/off control of the heating chamber so that the heating chamber is operated parallel to the machine.

In the next section you will program a query of the temperature values and integrate it in the program as condition for control of the conveyor.

4.7 Integrate temperature comparison as condition in control program

Introduction

The following steps will show you how to evaluate the temperature values of the PID controller and integrate them as condition for control of the machine. The pasteurization process is controlled by the following organization blocks:

- Main [OB1]

  The program for controlling the machine dependent on the temperature values of the PID controller is executed in the "Main [OB1]" organization block.

- PID [OB200]

  The technological object "PID_Compact" of the PID controller and the simulation block "PROC_C" are executed in the organization block "PID [OB200]".

Connection between PID controller and control program

The temperature monitoring of the PID controller is integrated as follows into the program of the organization block "Main [OB1]":

The actual temperature value of the "temperature" tag, which is used as input for the PID controller, should be read by the program in the organization block "Main (OB1)" and compared with a minimum and maximum value.

- If the actual temperature value is between a minimum value of 73°C and a maximum value of 77°C (setpoint +/- 2°C), the conveyor will be driven.
- If the minimum value of 73°C has not been reached yet or the maximum value of 77°C has been exceeded, the "conveyor_drive_ON_OFF" tag will be reset.

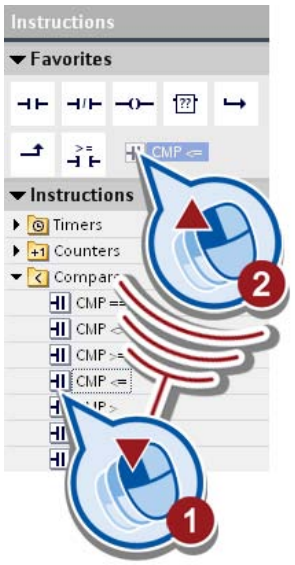
Requirements

- The organization block "Main [OB1]" is open.

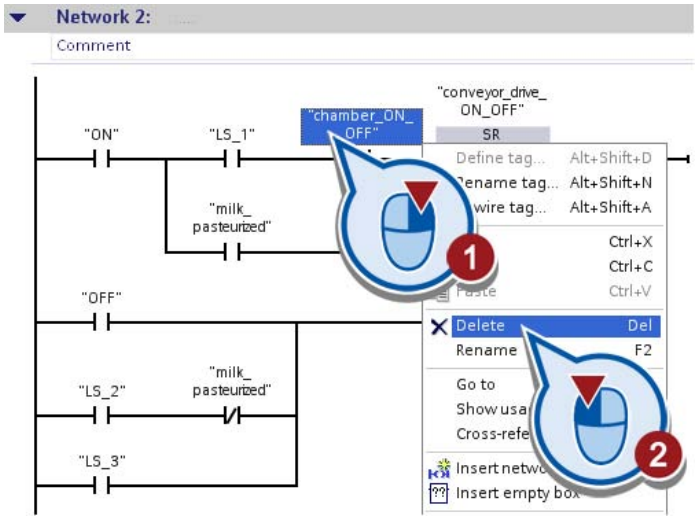
Procedure

To insert a value comparison of the "temperature" tag as condition for driving the conveyor, follow these steps:

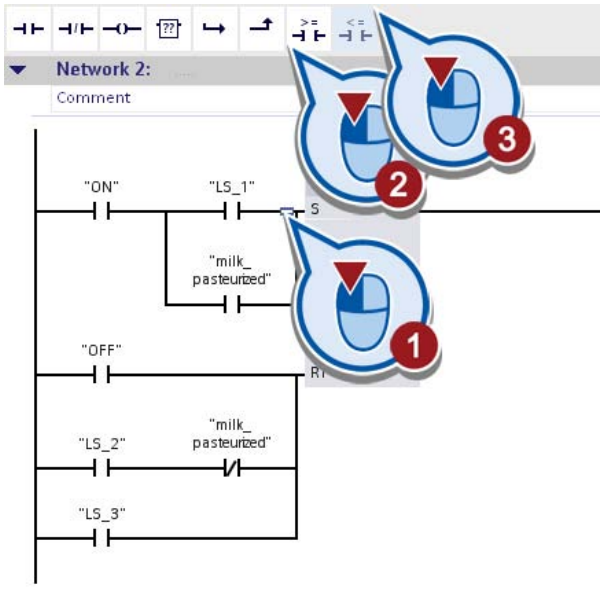
1. Insert the instructions "'Greater than or equal to (CMP>=)" and "Less than or equal to (CMP<=)" in the favorites list.



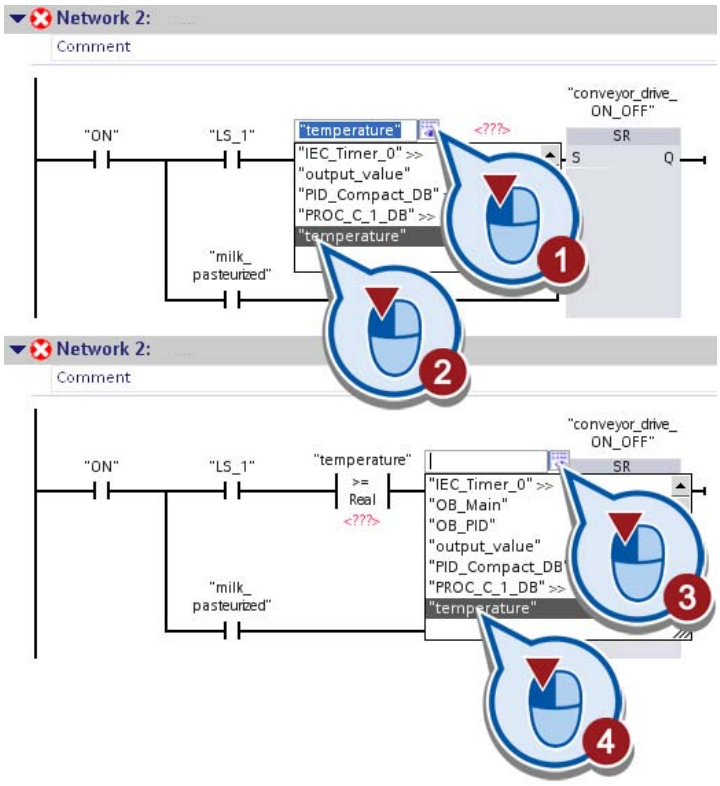
2. Open the second network of the organization block "Main [OB1]".
3. Delete the normally closed contact with the "chamber_ON_OFF" tag.



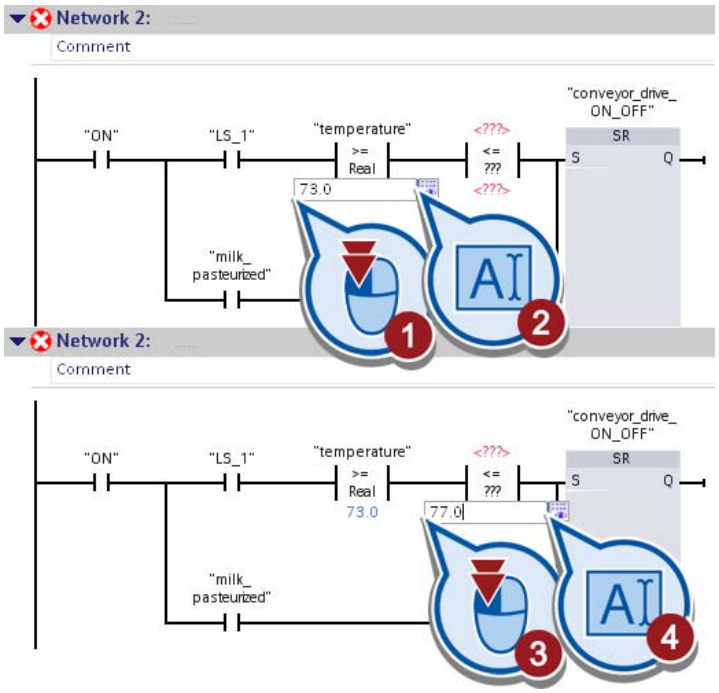
4. Insert the instructions "'Greater than or equal to (CMP>=)" and "Less than or equal to (CMP<=)" from the favorites list.



5. Interconnect the instructions "'Greater than or equal to (CMP>=)" and "Less than or equal to (CMP<=)", respectively, with the "temperature" tag.



6. In the operand placeholder beneath the instruction "Greater than or equal to" enter the value "73.0" and in the operand placeholder beneath the instruction "Less than or equal to" the value "77.0".



7. Click the "Save project" button on the toolbar to save the project.

Result

You have programmed the temperature value as condition for driving the conveyor. The conveyor is set into motion when the machine is in operation and at least one of the following conditions is satisfied:

- The light barrier "LS1" is activated and the temperature in the heating chamber is between 73°C and 77°C. This means:
  - The "LS_1" tag has the signal state "1".
  - The value of the "temperature" tag is between "73" and "77".
- The milk has been pasteurized. In this case the "milk_pasteurized" tag has the signal state "1".

In the next section you will adjust the link of the status light in the HMI screen.

4.8 Adjust HMI screen

Introduction

The following steps show you how to replace the status light of the heating chamber with an I/O field.

The heating chamber was switched on in the extended example, if the "chamber_ON_OFF" tag had the value "1". Because the temperature is now controlled by means of the PID controller, this PLC tag is no longer used. Instead of the status light, the current temperature should now be displayed in an I/O field.

I/O fields

Process values can be entered or displayed in the HMI screen by means of an I/O field.

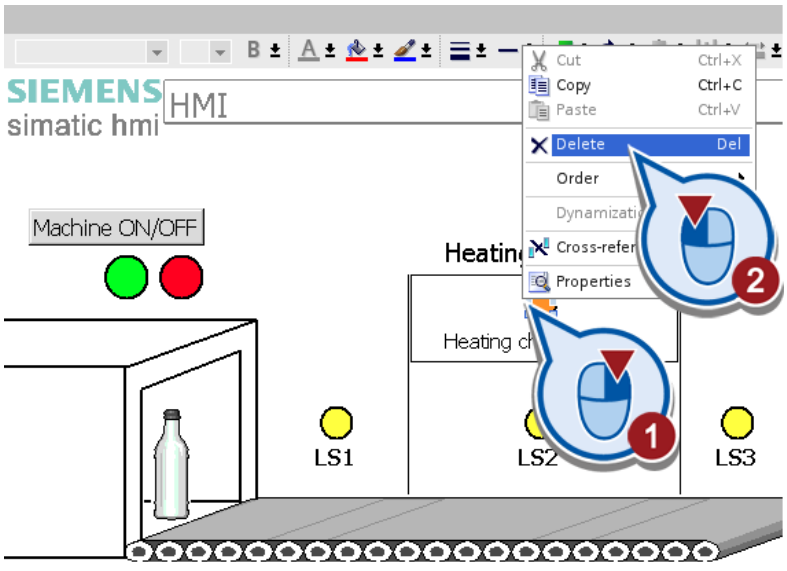
Requirements

- The program has been created.
- The HMI screen is open in the editor.

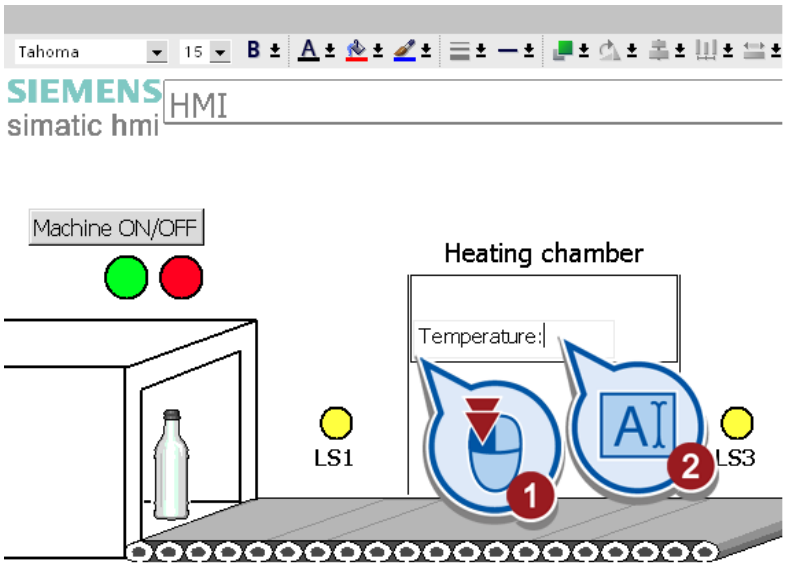
Procedure

To display the temperature value in the HMI screen, follow these steps:

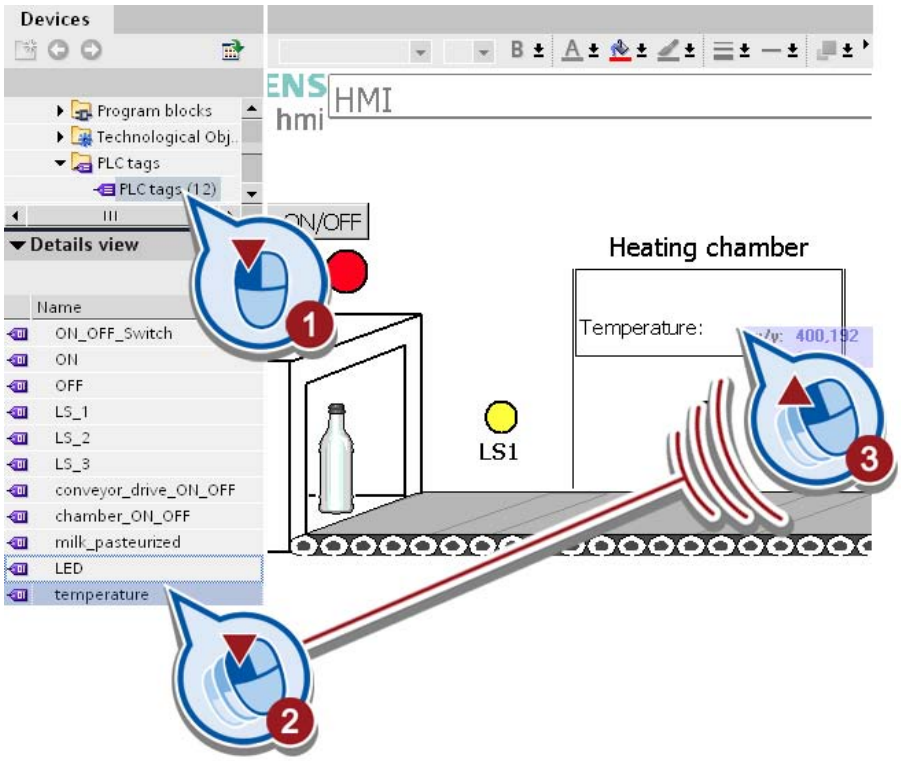
1. Delete the orange colored circle.



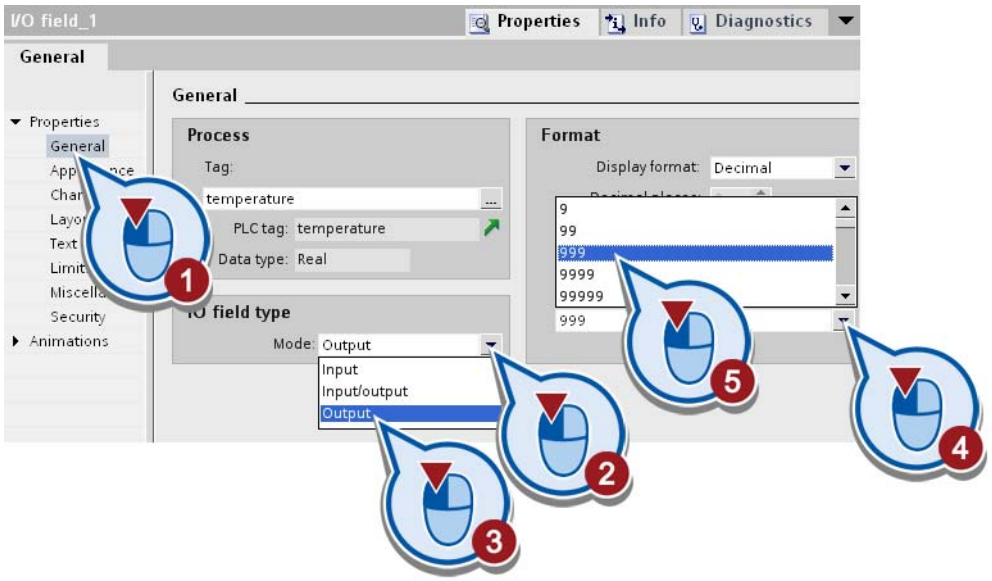
2. Change the content of the text block to "Temperature".



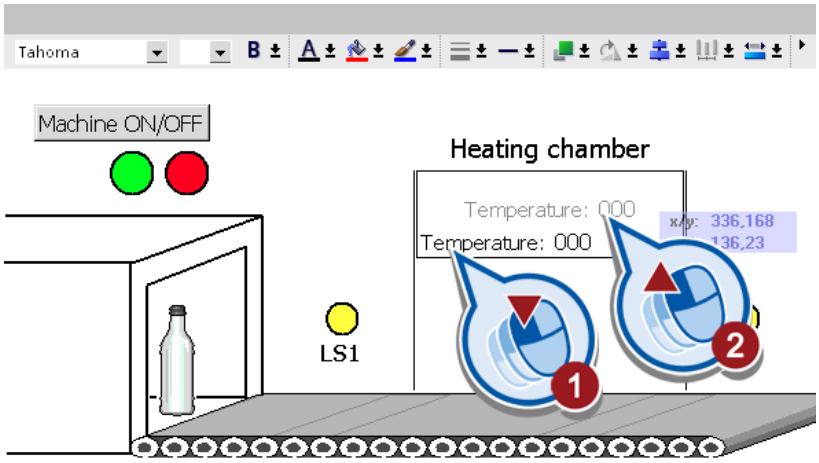
3. Insert an I/O field in the HMI screen by means of the details view of the PLC tag.



4. Change the field type and the format of the I/O field.



5. Adjust the size and position of the text and I/O field.



6. Click the "Save project" button on the toolbar to save the project.

Result

You have created an output field for the value of the "temperature" tag. When the user program is executed the current value of the tag is displayed in the HMI device.

4.9 Active PID controller in online mode

Introduction

The following steps show you how to simulate the function of the PID controller. To do this, first load the program in the PLC and activate the PID controller in online mode. Observe the ongoing control in the trend window after optimization of the controller.

Controller optimization

Use the optimization step to adapt the controller to the controlled system. You have two options to optimize the controller:

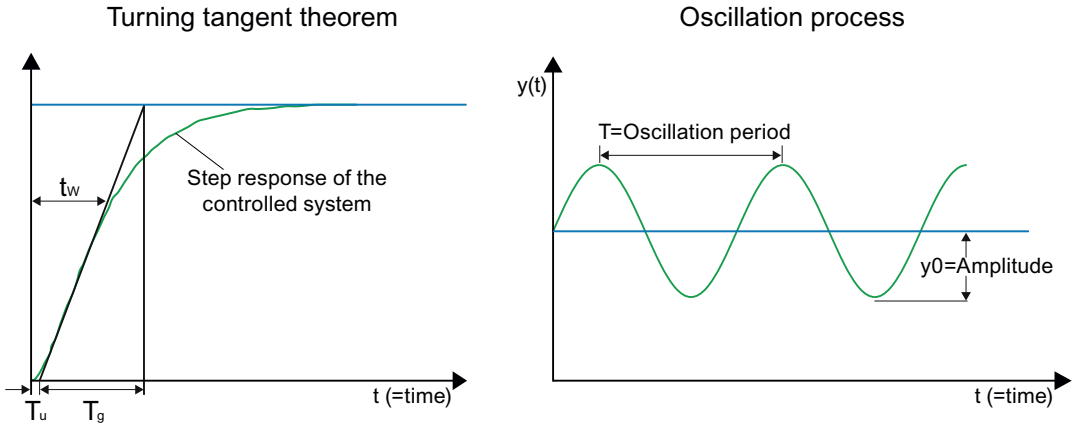
- Startup tuning

  The turning tangent theorem is applied in startup tuning. The theorem determines the time constant of the step response. A turning point is present in the step response of the controlled system. A tangent will be applied to the turning point; you use this tangent to determine the process parameters delay time (T_u) and recovery time (T_g). The optimized controller parameters will be determined based on these process parameters. The distance between set value and actual value must be at least 30% to determine the parameters with the turning tangent theorem. If this is not the case, the controller parameters will be determined automatically with the oscillation process and the function "Tuning in run".

- Tuning in run

  Tuning in run uses the oscillation process to optimize the controller parameters. You use this process to indirectly determine the behavior of the controlled system. The gain factor will be increased until it reaches the stability limit and the controlled variable oscillates evenly. The controller parameters are calculated based on the oscillation period.

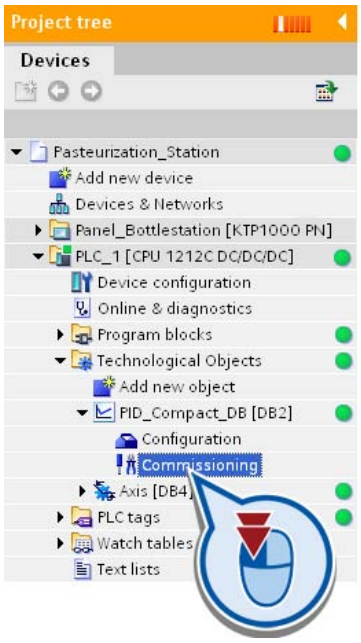
The following figure shows the step response of the controlled system with the turning tangent theorem (startup tuning) and during the oscillation process (tuning in run):



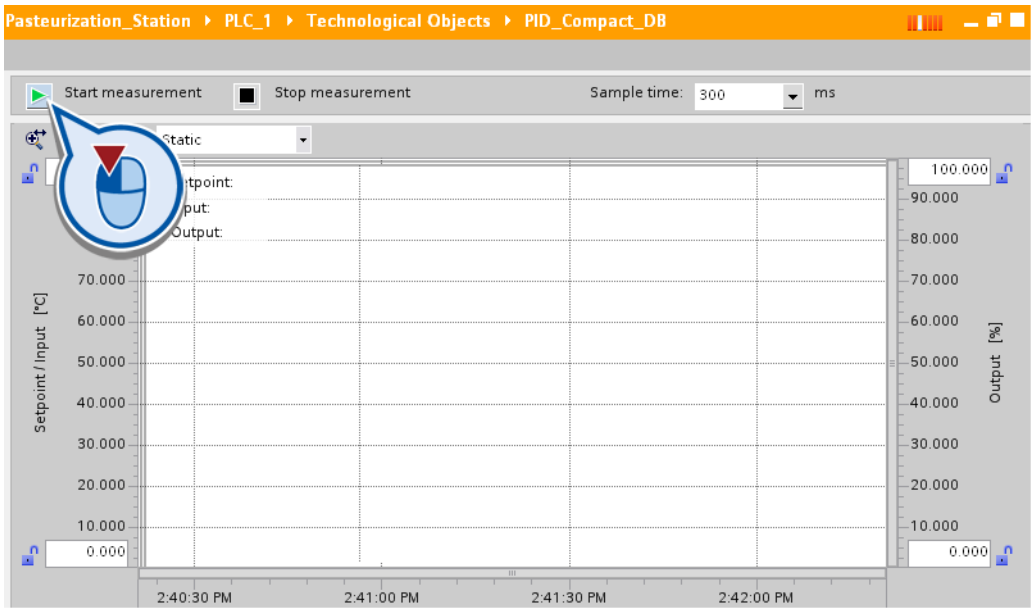
Procedure

To activate the PID controller and start the simulation, follow these steps:

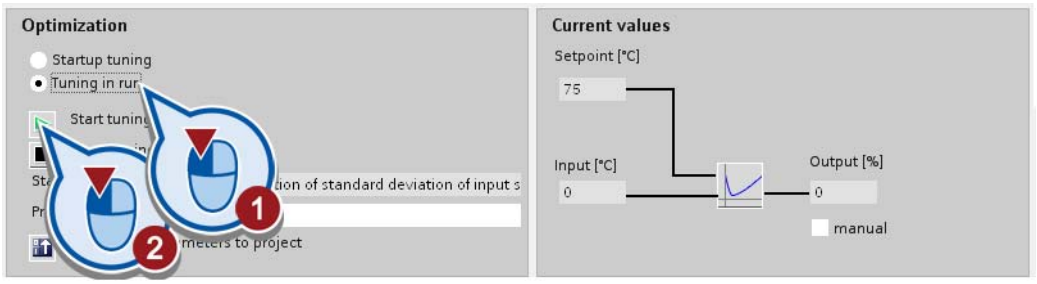
1. Load the program to the PLC and activate the online connection. For additional information see the "Loading the program to the target system (Page 48)" section.
2. Start the commissioning from the project tree.



3. Enable the functions of the commissioning window.

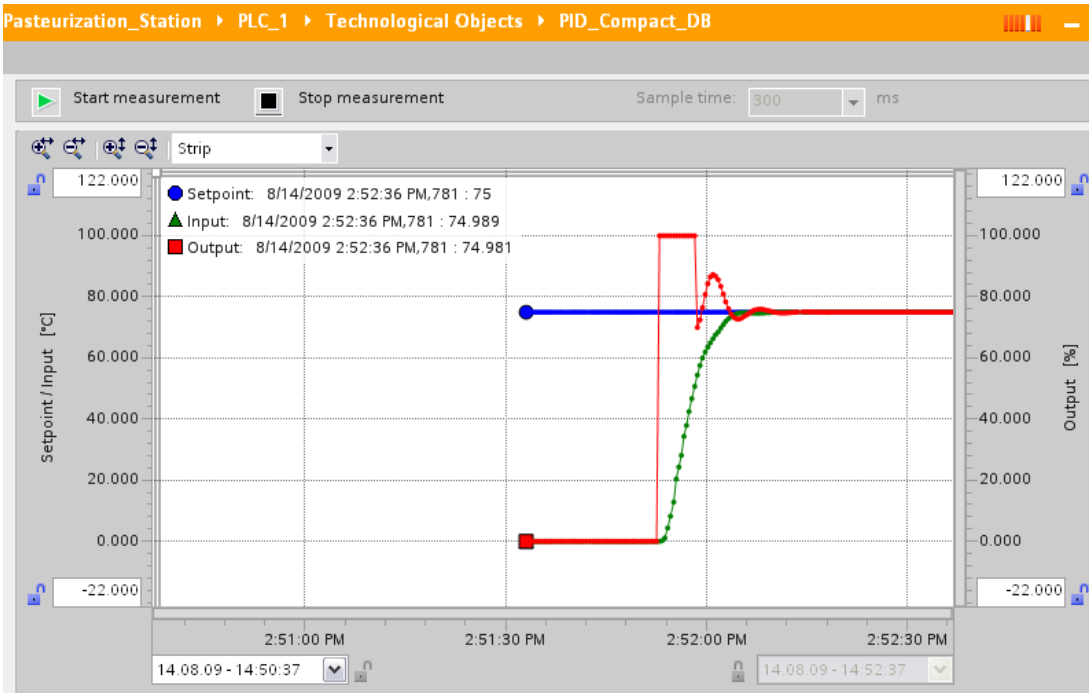


4. Perform a fine-tuning of the controller optimization, by starting the tuning in run.



The self-tuning starts. The current work steps and any errors that occur are shown in the "Status" field. The progress bar shows the progress of the current work step. Several steps are required for controller optimization. The progress display will start at "0" for each step. If the self-tuning was completed without error message, the PID parameters have been optimized. The PID controller switches to automatic mode and uses the optimized parameters.

5. Monitor the curve shape in the top section of the "Commissioning" window.



Note

Shape of the curve

In time the shape of the curve gets closer and closer to the setpoint. To monitor the complete shape of the curve, use the vertical scroll bar in the bottom area of the graphic display.

After the controller has switched to automatic mode, the controller reacts to the temperature changes simulated by the "PROC_C" block by automatically adjusting the controlled variable.

Result

You have activated the PID controller in online mode. The PID parameters that were optimized prior to the start of automatic mode are retained during POWER ON and restart of the CPU. If you want to reuse the PID parameters optimized in the CPU when you load the project data to the CPU once again, you can save the PID parameters in the project.

Example "Motion"

5

5.1 Introduction

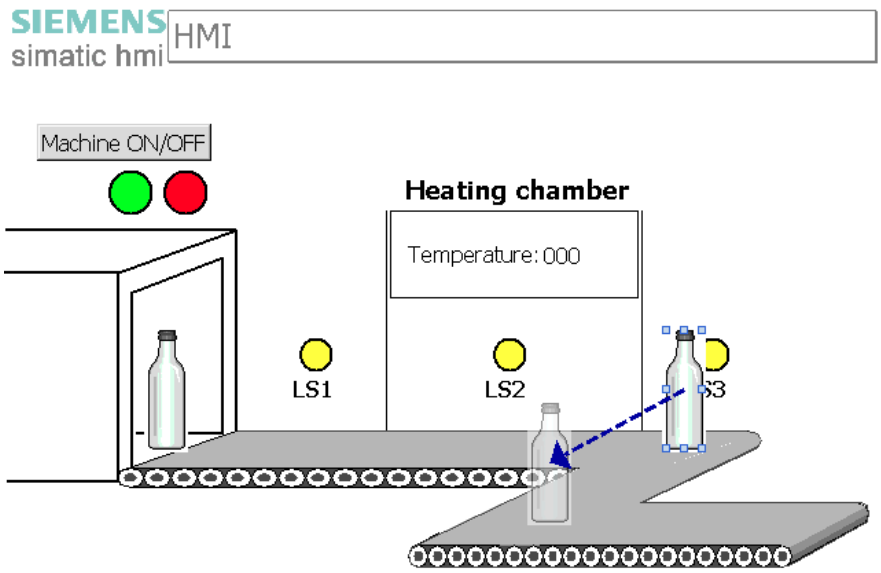
Loading a project

If you have skipped the previous chapters, you can load the project status at the end of the last chapter by opening the project "Extendet_Example_PID". The result at the end of this chapter is also stored as project under the name "Extendet_Example_Motion".

For additional information on loading of projects, see the "Loading a project (Page 17)" section.

Use of motion

In this chapter you add a conveyor to the example. Starting from the position of the light barrier "LS3", the bottle is transported via a connection piece to an additional conveyor.



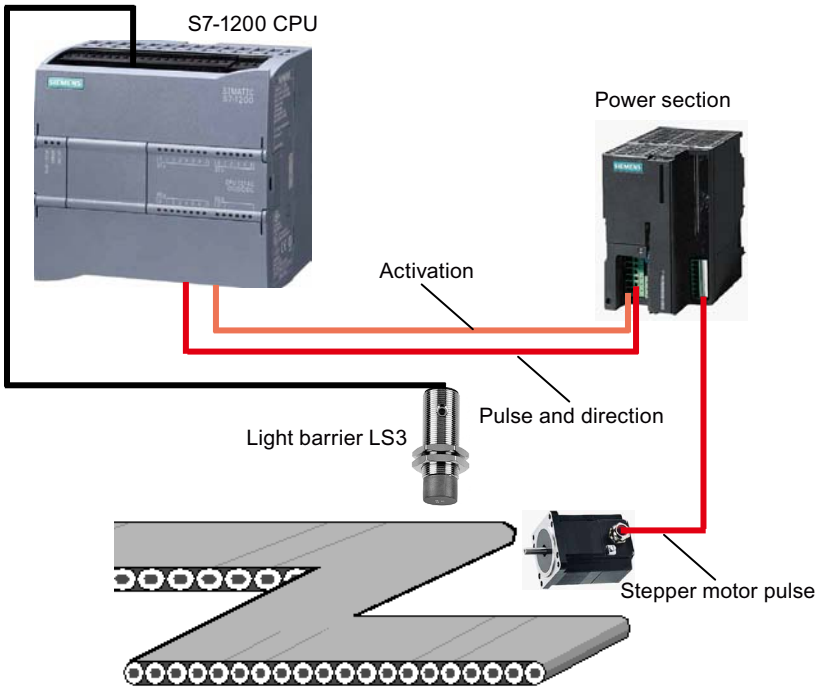
Use the technological object "Axis" to control the connection between the two conveyors. The PLC is integrated into the project as follows:

- If the machine is switched on, the axis for activating a stepper motor is enabled on the connection piece.
- When the light barrier "LS3" is activated, the axis of the motor is triggered.
- The bottle is moved to the second conveyor relative to the position of the light barrier "LS3".
- When the position is reached, the second conveyor is activated.

Hardware configuration

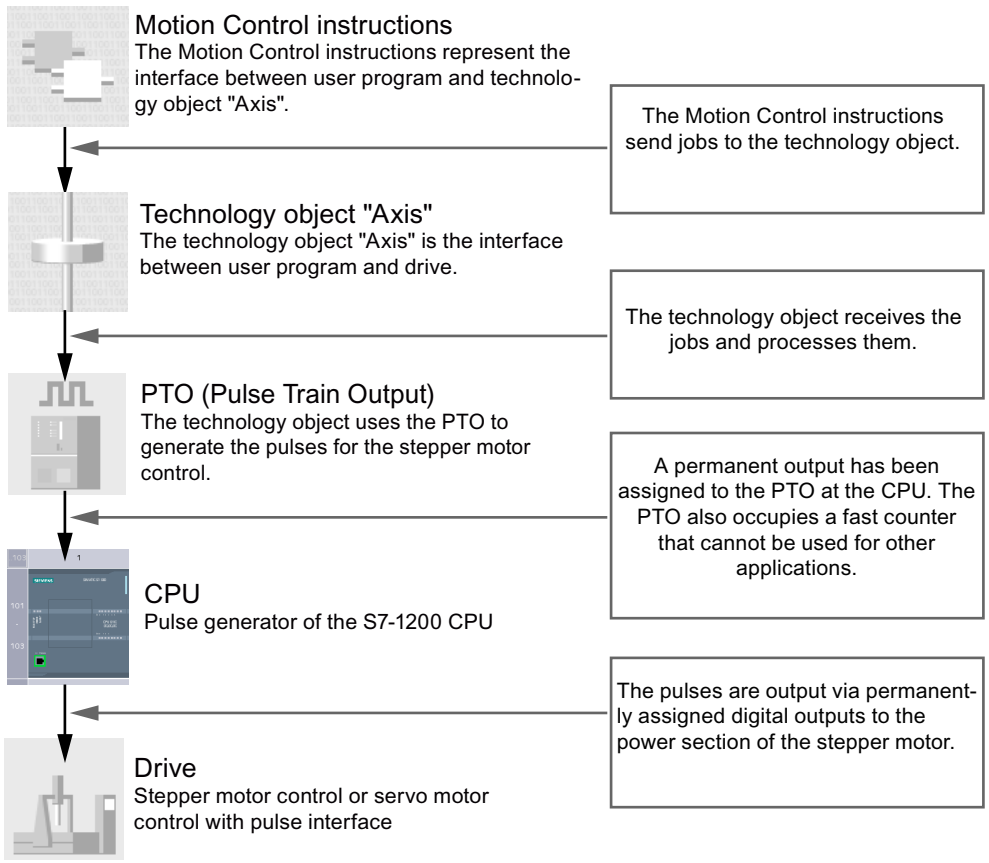
To implement the example, you require no hardware other than the PLC. If a PLC with relay outputs is used, a signal board has to be used for the outputs of the PTO. The PTO (Pulse Train Output) generates quick pulse trains via the outputs, which are used to control the motion control operations.

The following figure shows how the possible hardware configuration could look for this example:



Sequence of the pulse generation

The following figure describes the sequence of the pulse generation via motion control instructions.


Motion Control instructions
The Motion Control instructions represent the interface between user program and technology object "Axis".
The Motion Control instructions send jobs to the technology object.
Technology object "Axis"
The technology object "Axis" is the interface between user program and drive.
The technology object receives the jobs and processes them.
PTO (Pulse Train Output)
The technology object uses the PTO to generate the pulses for the stepper motor control.
A permanent output has been assigned to the PTO at the CPU. The PTO also occupies a fast counter that cannot be used for other applications.
CPU
Pulse generator of the S7-1200 CPU
The pulses are output via permanently assigned digital outputs to the power section of the stepper motor.
Drive
Stepper motor control or servo motor control with pulse interface

After the pulses for stepper motor control have been generated via the outputs and were output to the power section of the stepper motor, the pulses will be converted into an axial motion by the power section of the stepper motor. The connection between the conveyors is driven by the axis.

Steps

In this chapter you will perform the following steps:

- You create the technological object "Axis".
- You configure the technological object "Axis", by:
  - assigning a pulse generator to the technological object.
  - Configuring the pulse generator as PTO (Pulse Train Output).
  - Assigning a high-speed counter to the pulse generator (is automatically selected).
- You create two motion control instructions:
  - One to enable the axis (MC_Power).
  - One for relative positioning of the position of the light barrier "LS3" (MC_MoveRelative).

5.2 Insert technological object "Axis"

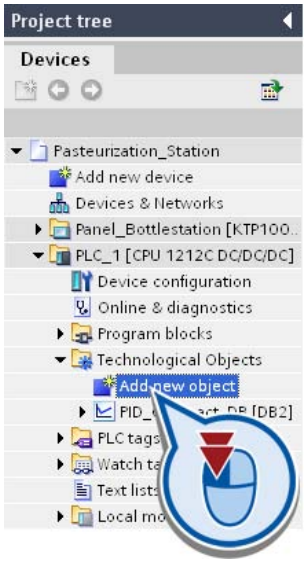
Introduction

The following steps show you how to create a new technological object "Axis".

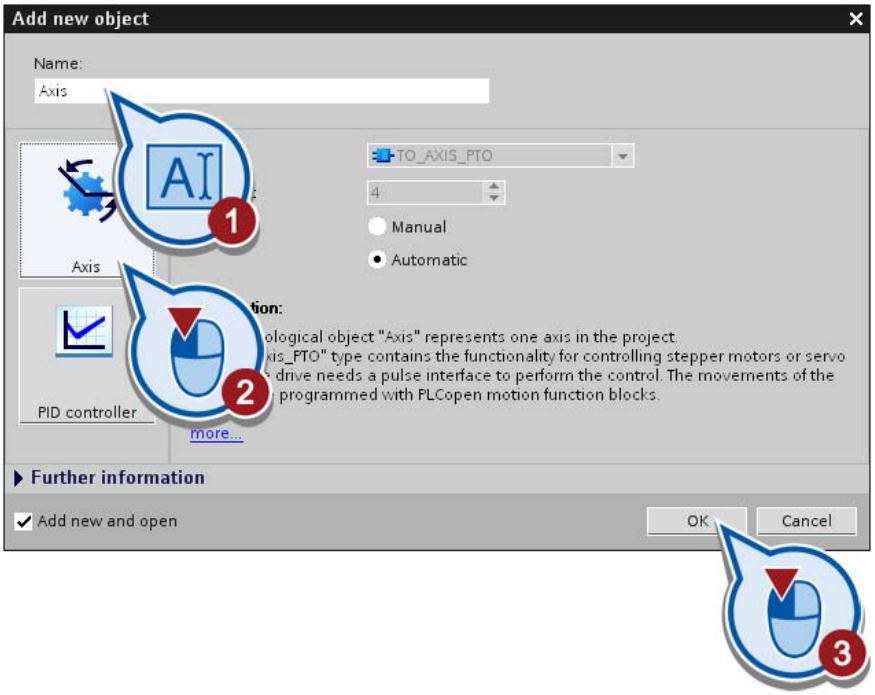
Procedure

To create the technological object "Axis", follow these steps:

1. Create a new technological object in the project tree.



2. Create a new object "Axis".



The configuration of the technological object is started in the work area.

Result

You have created the technological object "Axis". This is stored in the "Technological objects" folder in the project tree.

In the next section you will configure the created technological object.

5.3 Configure technological object "Axis"

Introduction

The following steps show you how to configure the technological object "Axis".

Requirements

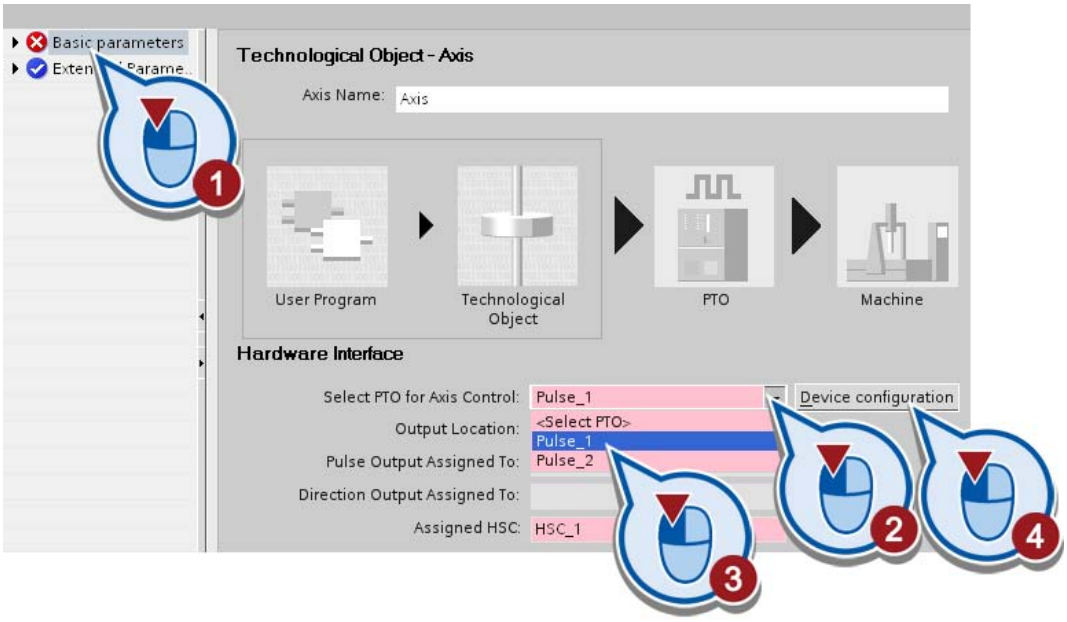
- The technological object "Axis" has been created.
- The configuration of the technological object has been started.

  If the configuration was not automatically opened, start the configuration from the project tree.

Procedure

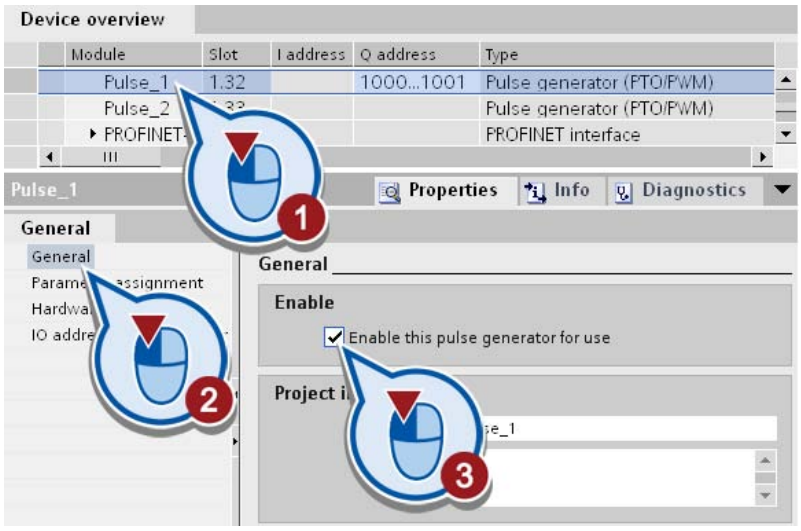
To configure the drive control, follow these steps:

1. Select the pulse output "Pulse_1" to the PTO drive control and switch to the device configuration.

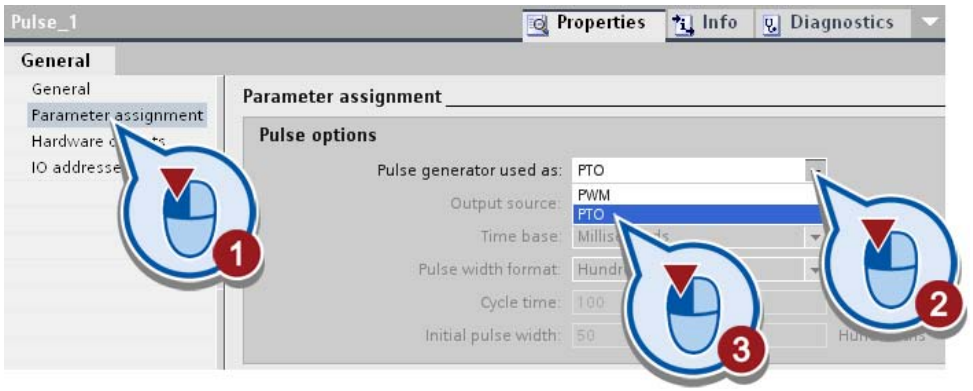


The high-speed counter "HSC_1" is automatically assigned during selection of "Pulse_1". The device view opens.

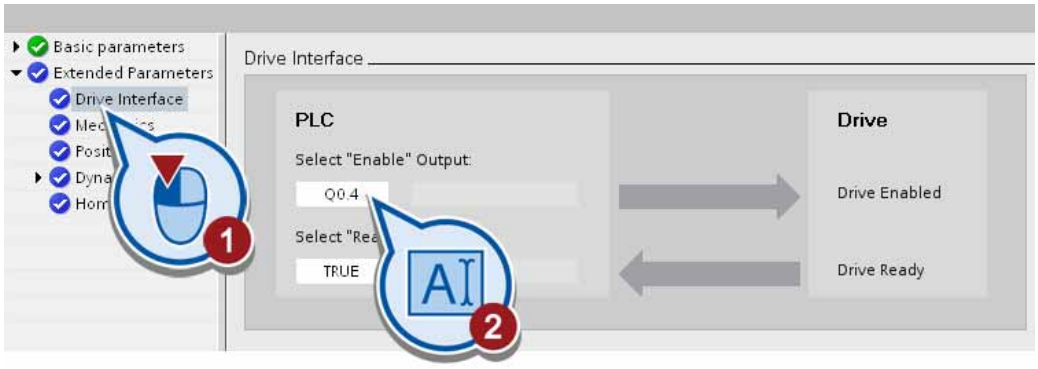
2. Activate the pulse generator.



3. Configure the pulse generator as PTO using parameters.



4. Switch to the configuration editor of the technology object "Axis". Specify the PLC output over which the drive enable is to take place.



5. Click the "Save project" button on the toolbar to save the project.

Result

You have assigned the created technological object "Axis" to the pulse generator "Pulse_1" and parameterized it as PTO. The high-speed counter "HSC 1" was automatically activated. The pulses that are output via the pulse generator are counted by means of the high-speed counter.

If you have correctly configured the technological object, the status of the "Basic parameters" area and the status of the "Extended parameters" is displayed in green in the configuration window.

5.4 Enable axis

Introduction

The following steps show you how to enable the motor axis of the conveyor.

To enable or disable an axis, use the motor control instruction "MC_Power". The instruction must be called once for each axis in the program.

The axis is centrally enabled or disabled by means of the motion control instruction ""MC_Power":

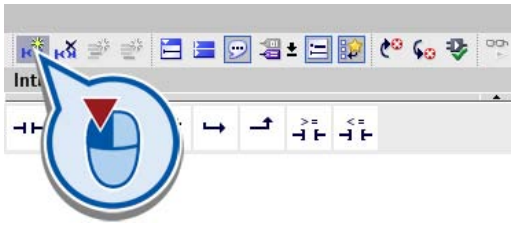
- If the axis is enabled, the enable is valid for all motion control instructions that were assigned to this axis.
- If the axis is disabled, all other motion control instructions for this axis have no effect. All current jobs will be interrupted.

Requirements

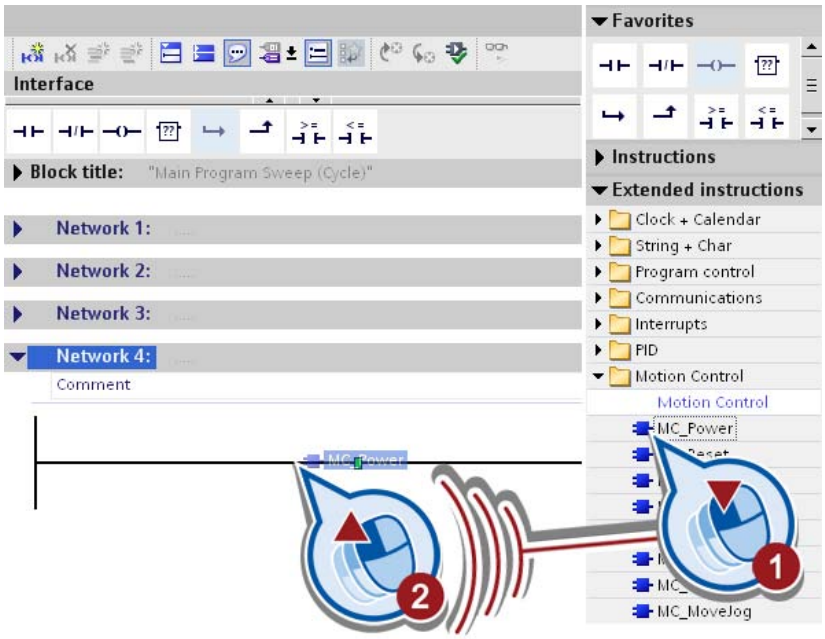
- The technological object has been configured correctly.
- The organization block "Main [OB1]" is open.

Procedure

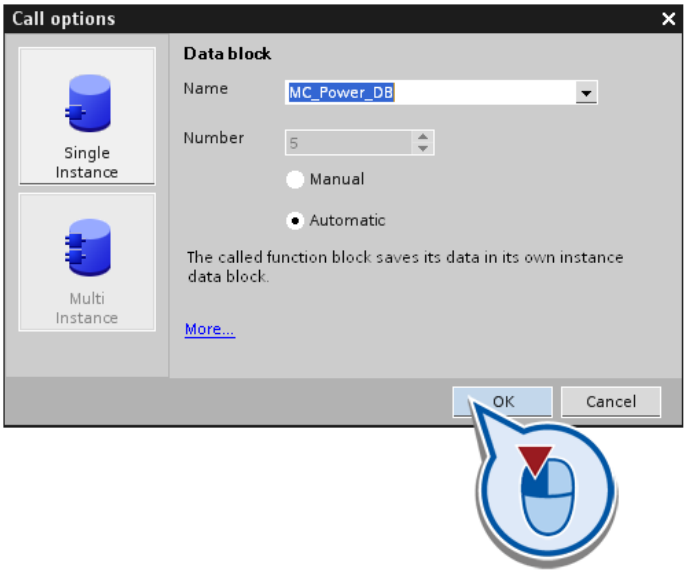
1. Create a new network in the organization block "Main [OB1]". A new network is inserted automatically as soon as you create an element in an empty network. If an empty network is already available, you can skip this step.



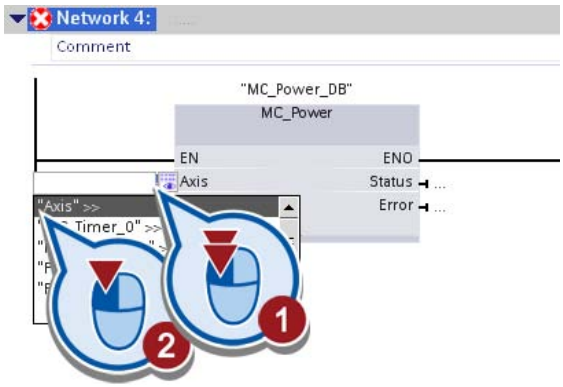
2. Create the motion control block "MC_Power" in the new network.



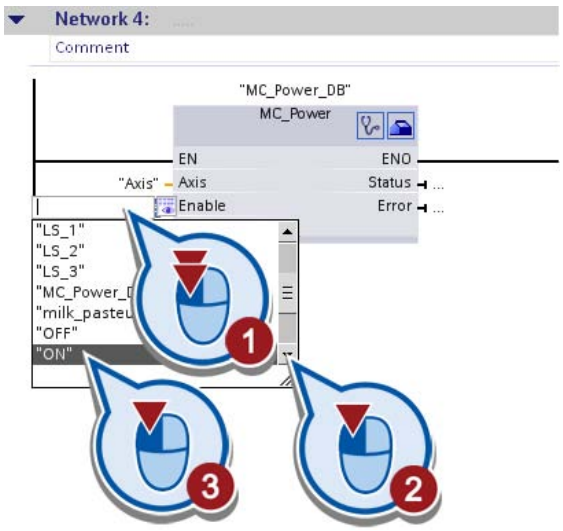
3. Confirm the creation of the new data block.



4. At the "Axis" input, select the previously configured axis "Axis".



5. At the "Enable" input, select the "ON" tag.



6. Save the project.

Result

You have inserted the instruction "MC_Power" to enable the axis in the program and assigned it to the technological object "Axis".

The enabling of the axis depends on the value of the "ON" tag at the "Enable" input:

- When the bit of the "ON" tag has the value "0" (machine off), the axis is disabled.
- When the bit of the "ON" tag has the value "1" (machine on), the axis is enabled.

In the next section you will program the motion of the conveyor relative to the starting point.

5.5 Position axis relative

Introduction

The following steps show you how to program the motion on the second conveyor relative to a starting position with the motion control instruction "MC_MoveRelative".

The motion is defined as follows:

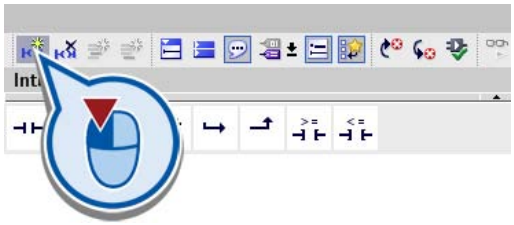
- The starting position of the motion is the position of the light barrier "LS3".
- The end position is 0.5 meter in positive direction on the axis between the first and second conveyor.
- When the end position is reached, the second conveyor is activated.

Requirements

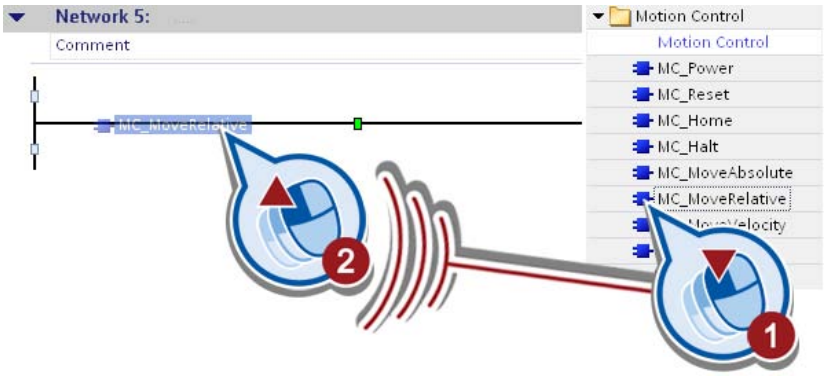
- The technological object "Axis" has been created and correctly configured.
- The motion control instruction "MC_Power" has been created in the organization block "Main [OB1]".
- The organization block "Main [OB1]" is open.

Procedure

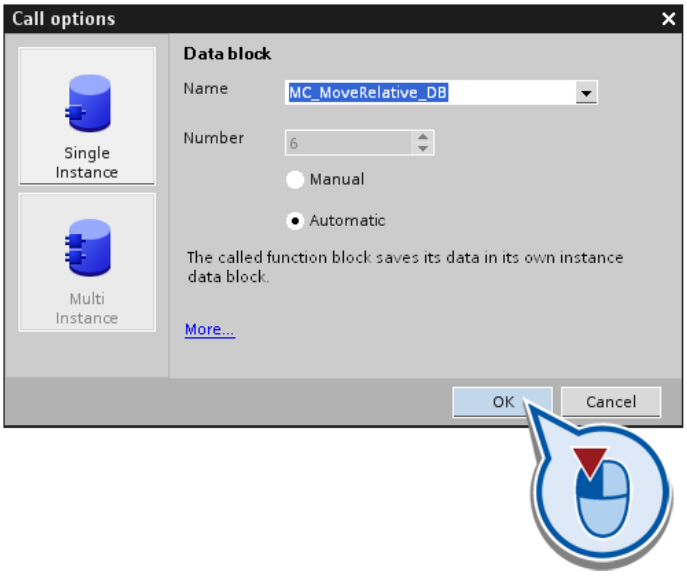
1. Create a new network in the organization block "Main [OB1]". A new network is inserted automatically as soon as you create an element in an empty network. If an empty network is already available, you can skip this step.



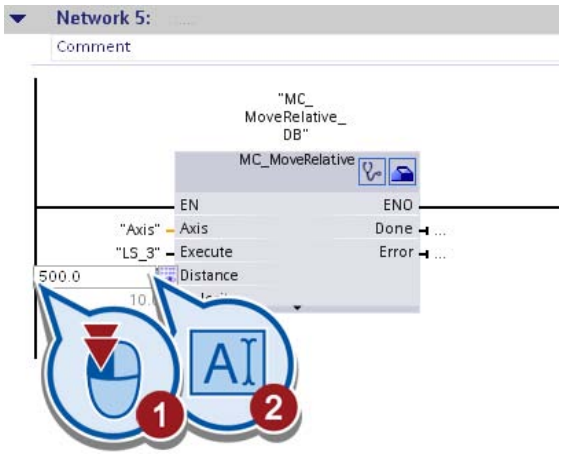
2. Create the block "MC_MoveRelative" in the new network.



3. In the "Call options" dialog box, click OK to confirm the creation of the new data block.

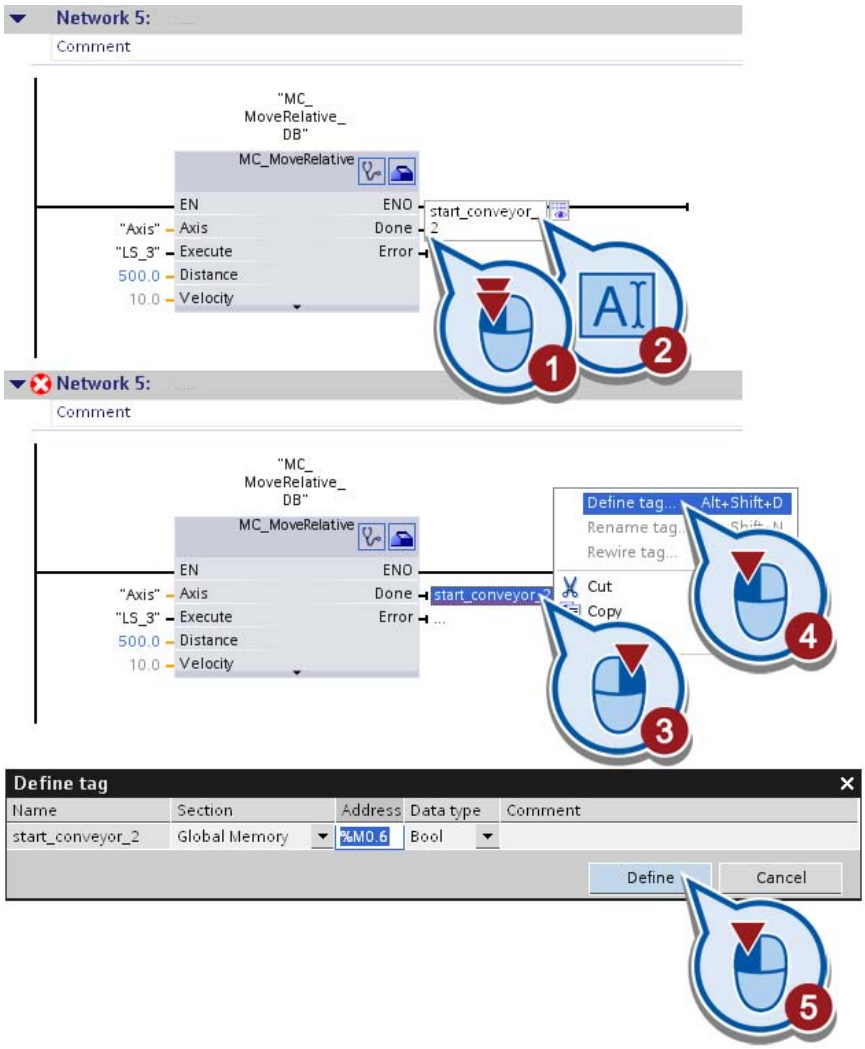


4. At the "Axis" input, select the box of the previously configured axis "Axis".
5. At the "Execute" input, select the box of the "LS_3" tag.
6. Enter the value "500.0" at the "Distance" input and confirm it with the Enter key.



The standard setting for the measurement unit of "Distance" is millimeters and was previously taken over in the configuration.

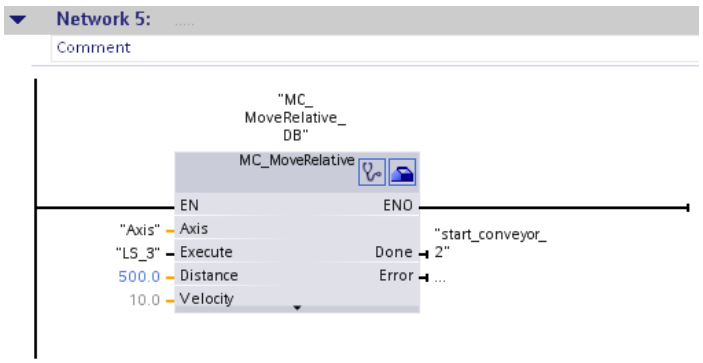
7. Create a new tag "start_conveyor_2" at the "Done" output.



8. Click the "Save project" button on the toolbar to save the project.

Result

You have programmed the motion of the conveyor relative to the position of the light barrier "LS3".



The motion is started when the light barrier "LS3" is activated. When the target position has been reached, the bit of the "start_conveyor_2" tag is set at "Done" output, by means of which the second conveyor can be set into motion.

In the next section you will add more elements to the HMI screen to visualize the execution of the programmed processes.

5.6 Expand HMI screen

5.6.1 Change graphic object conveyor

Introduction

The following steps show you how to replace the existing graphic object "Conveyor" with the graphic "ConveyorMotion.wmf" in the HMI screen. The graphic contains two conveyors instead of one; these are connected to each other.

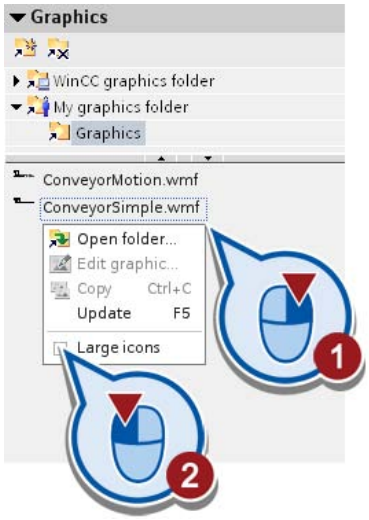
Requirements

The ZIP file "WinCC Graphics" has been extracted and stored in a local directory. For the directory, a link has been created in the "Toolbox" task card of the "Graphics" pane. See Graphic object "Conveyor" (Page 74) section.

Procedure

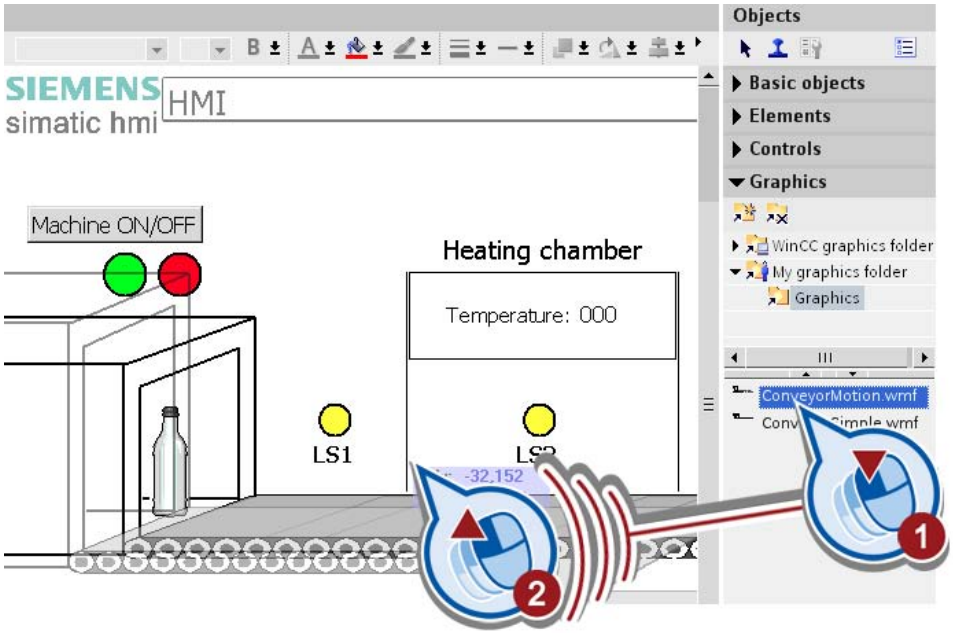
To expand the HMI screen, follow these steps:

1. Open the HMI screen.
2. Open the link to the local folder "WinCC Graphics" in the "Graphics" pane of the "Toolbox" task card.
3. Disable the "Large icons" option.

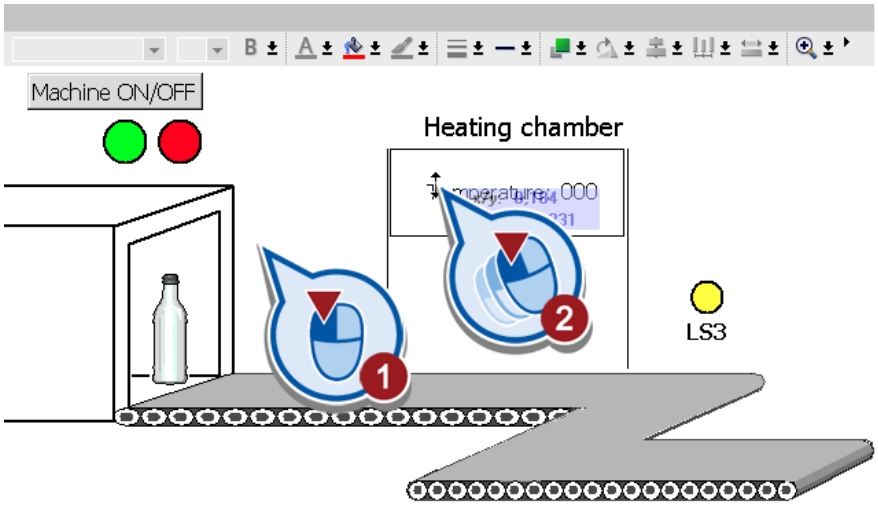


4. Drag the graphic object "ConveyorMotion.wmf" per drag-and-drop operation to the editor area, to replace the existing conveyor.

   Make sure that an arrow symbol is displayed next to the cursor when you insert the object. If the cursor is displayed with a Plus symbol, the graphic will be added, not replaced.



5. If necessary, scale the graphic object.



Result

You have replaced the graphic object "Conveyor".

In the next section you will add a second animated graphic object to the HMI screen.

5.6.2 Create second graphic object bottle

Introduction

In the following section you will create a second bottle with a movement animation from the first to the second conveyor.

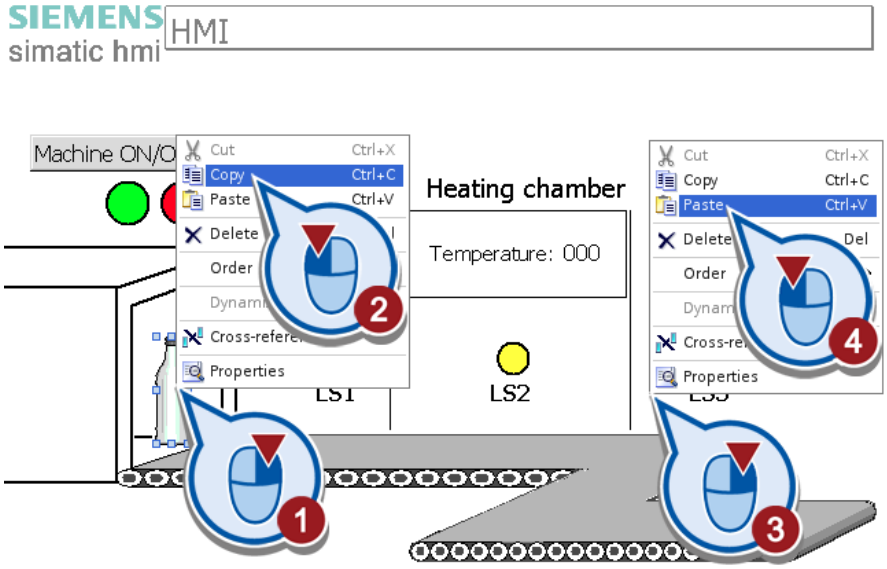
Requirements

The HMI screen is open.

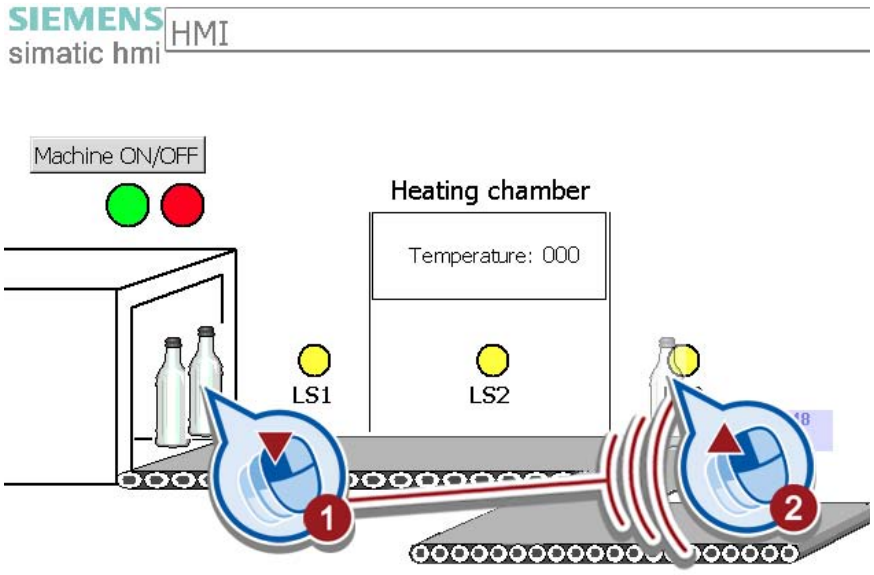
Procedure

To animate the second bottle in the HMI screen, follow these steps:

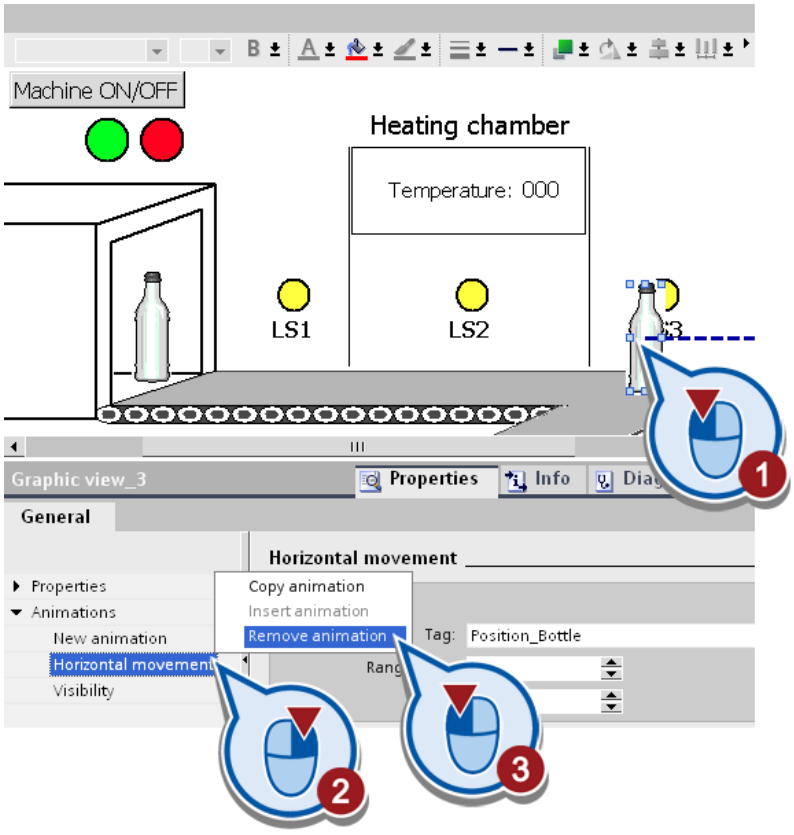
1. Copy the existing bottle.



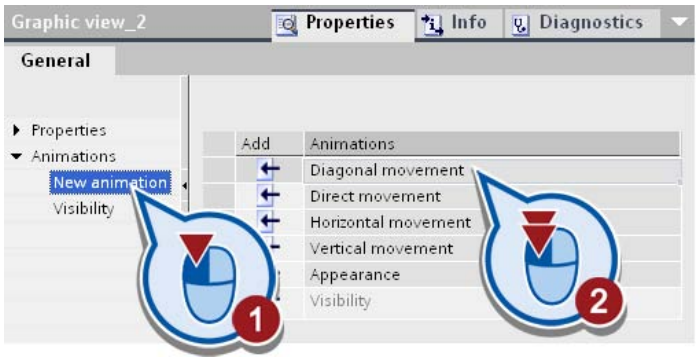
2. Position the copied bottle at the end of the movement animation of the first bottle in front of the light barrier "LS3".



3. Delete the "Horizontal movement" animation of the copied bottle.

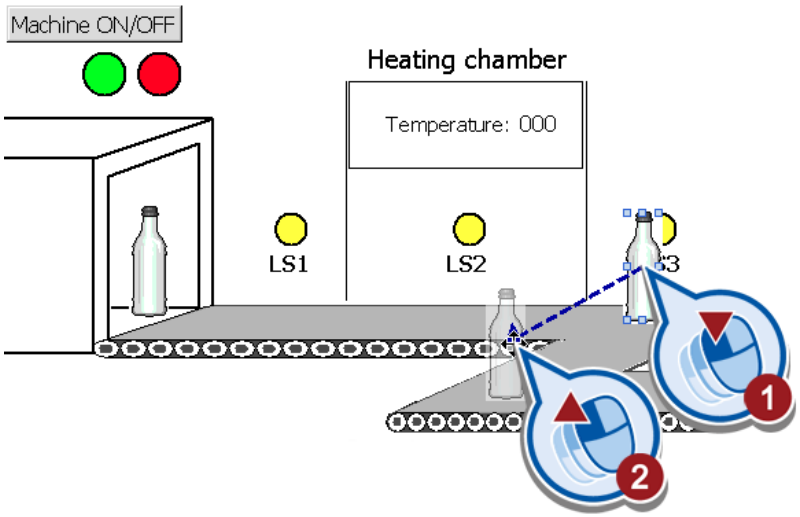


4. Create a "Diagonal movement" animation.

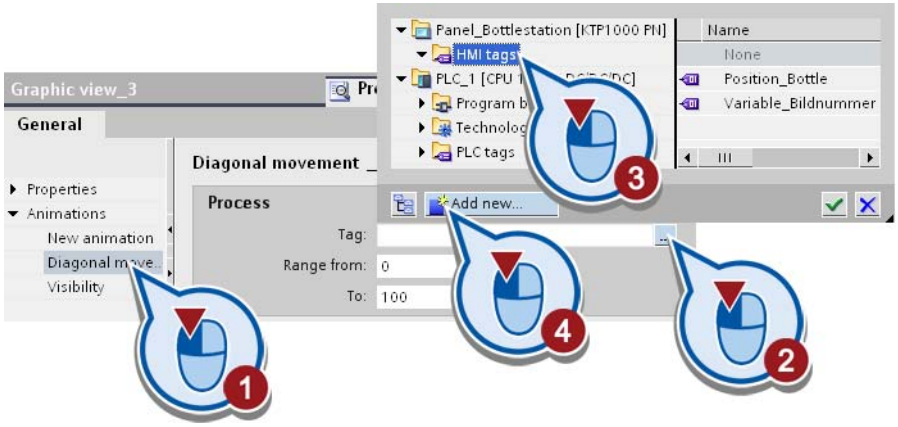


The motion animation is displayed with a blue arrow.

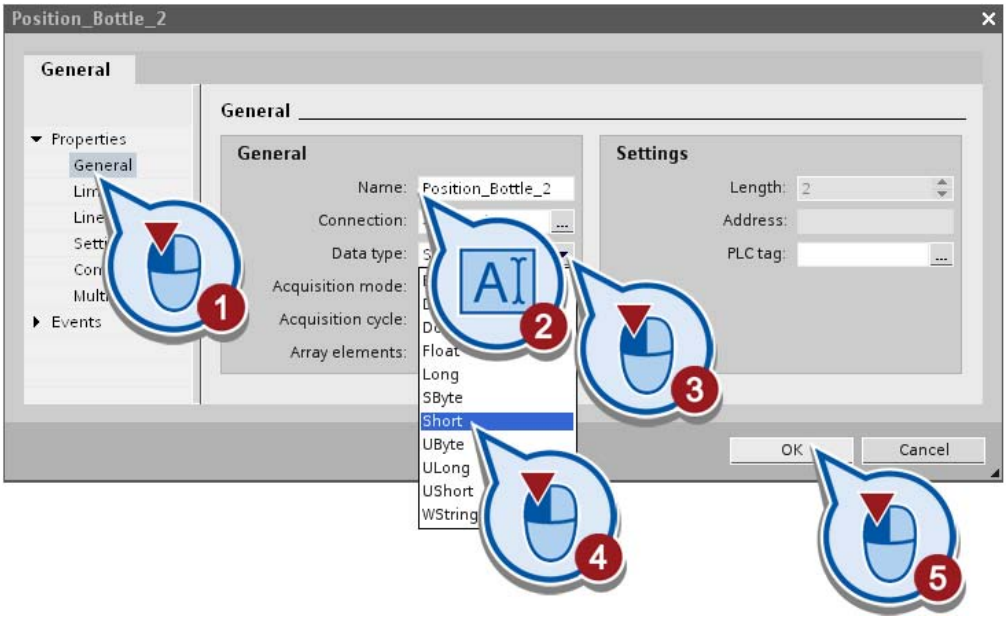
5. Drag the transparent representation of the bottle at its arrow tip to the second conveyor.



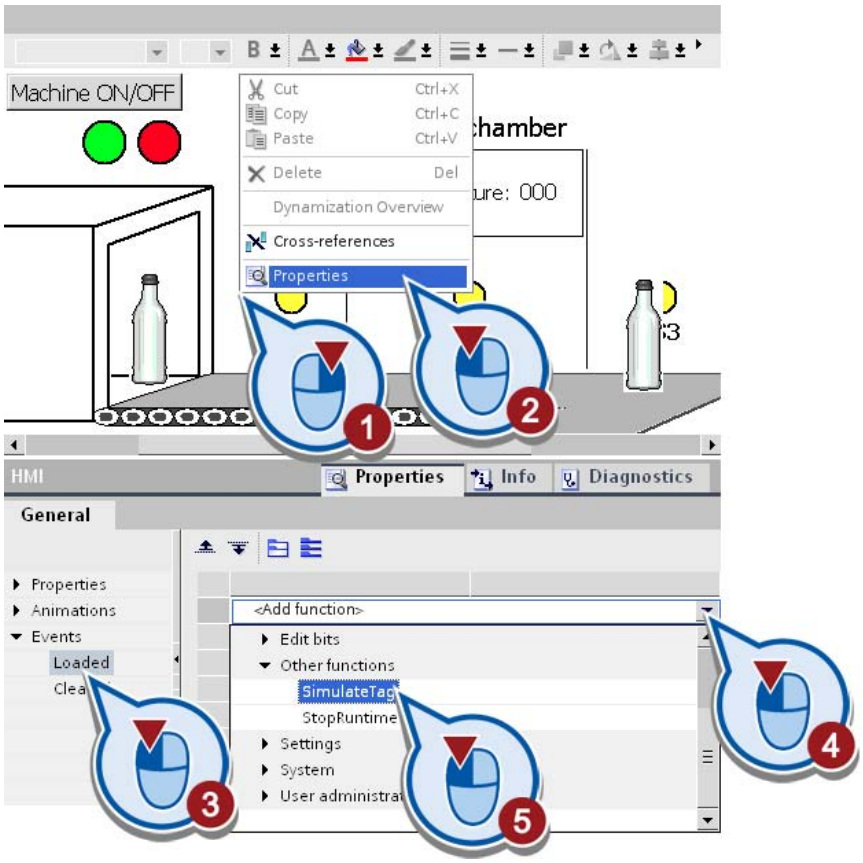
6. Create a new HMI tag for the diagonal movment animation.



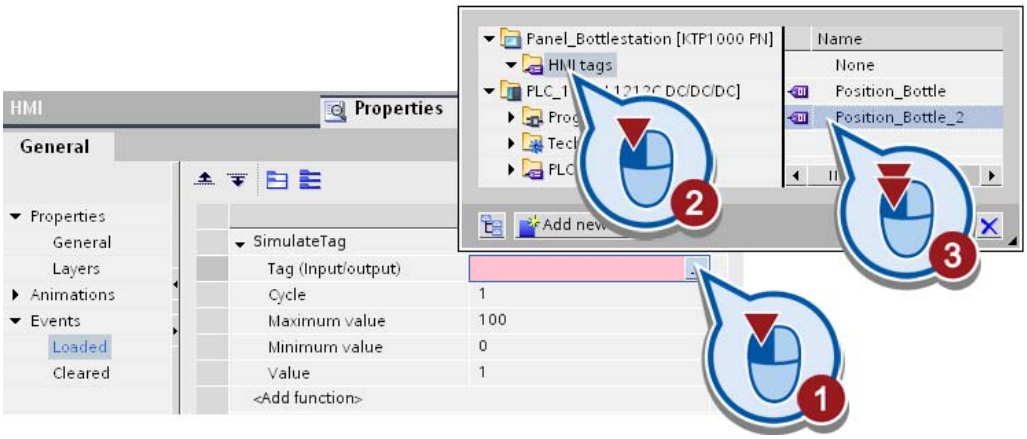
7. Use "Position_Bottle_2" as name and Short as data type.



8. Add the "SimulateTag" function to the event "Loaded" of the HMI screen.



9. Assign the "Position_Bottle_2" tag to the "Simulate Tag" function.



Result

You have created an animation that visualizes the motion of the second bottle from the first to the second conveyor.

In the next section you will change the visibility settings of the bottles on the conveyor.

5.6.3 Connect HMI objects with motion instruction

Introduction

The following steps show you how to animate the visibility of the bottles in the HMI screen dependent on the program progress.

For the visibility settings, use the signal state at the "Busy" parameter of the "MC_MoveRelative" instruction. The parameter contains information on whether the motion control instruction will be executed or not in that moment.

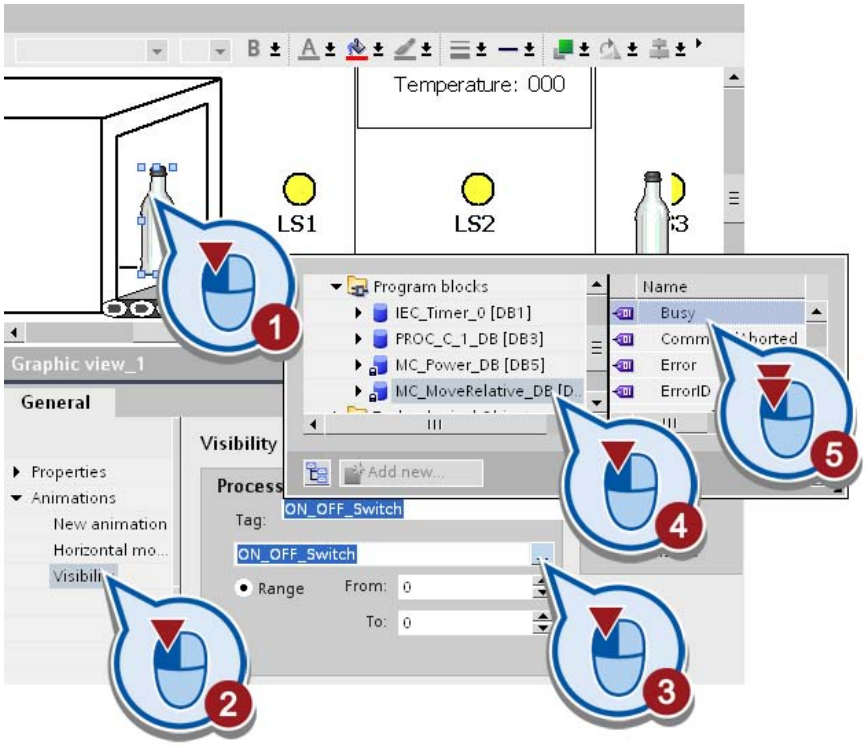
Requirements

- The "MC_MoveRelative" instruction has been inserted in the "Main [OB1]" organization block.
- The HMI screen is open.
- The motion animation has been created for the second bottle.

Procedure

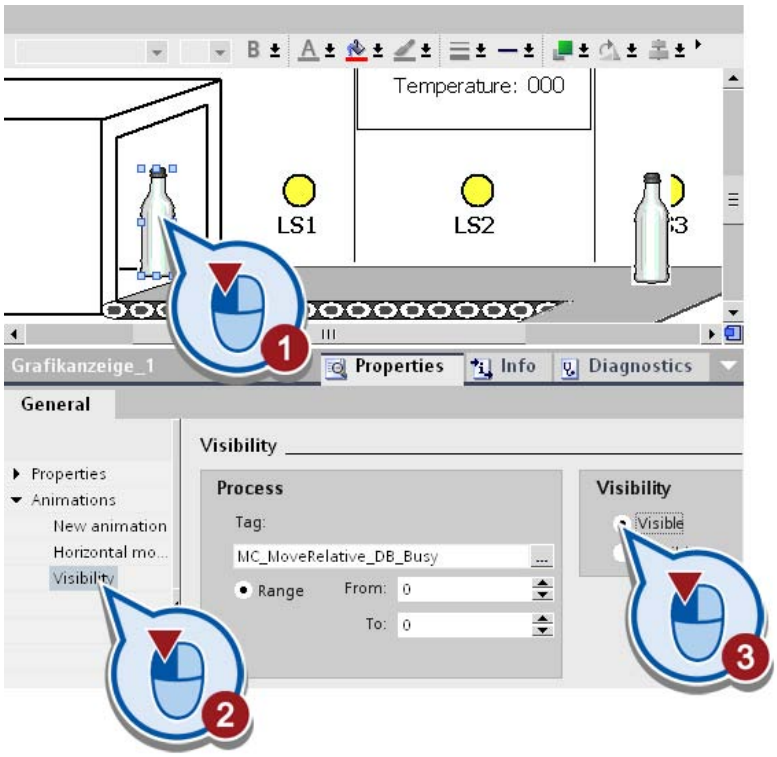
To configure the visibility of the bottles in the HMI screen, follow these steps:

1. Link the "Visibility" animation of the first bottle with the "Busy" parameter of the "MC_Move Relative" data block.



2. Link the "Visibility" animation of the second bottle with the "Busy" parameter of the "MC_Move Relative" data block.

3. Change the visibility of the first bottle for the range from "0" to "0" to "Visible".



Leave the visibility setting of the second bottle for the range "0" to "0" at "Invisible".

4. Click the "Save project" button on the toolbar to save the project.

Result

You have set the visibility of the bottles in the HMI screen depending on the signal state on the "Busy" parameter of the "MC_MoveRelative" instruction.

When the "MC_MoveRelative" instruction is executed, the second conveyor is set into motion. In this case the "Busy" parameter of the instruction has the signal state "1" and has the following effect on the HMI screen:

- The first bottle becomes invisible.
- The second bottle becomes visible and moves from the position of the third light barrier to the second conveyor.

If the "MC_MoveRelative" instruction is not executed, the "Busy" parameter of the instruction has the signal state "0" and has the follow effects on the HMI screen:

- The first bottle becomes visible and moves from the position of the first light barrier to the position of the third light barrier.
- The second bottle is no longer visible.

5.7 Simulate HMI screen

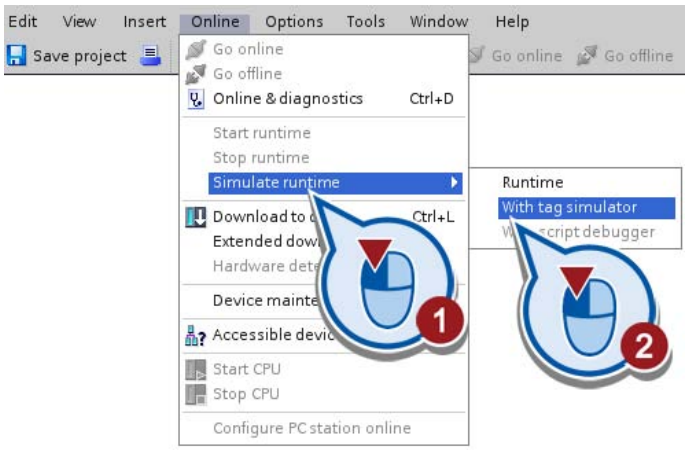
Introduction

The following steps show you how to test the created HMI screen with the Runtime Simulator. Use the Runtime Simulator to simulate the activation of the PLC input for the light barrier "LS3".

Procedure

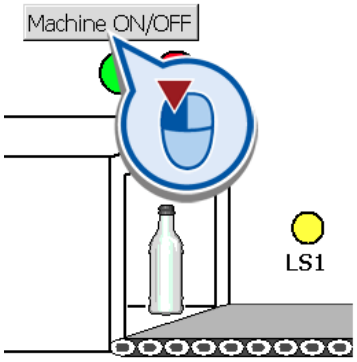
To start the simulation of the created HMI screen, follow these steps:

1. Open the HMI screen.
2. Start the runtime simulation via the menu bar.



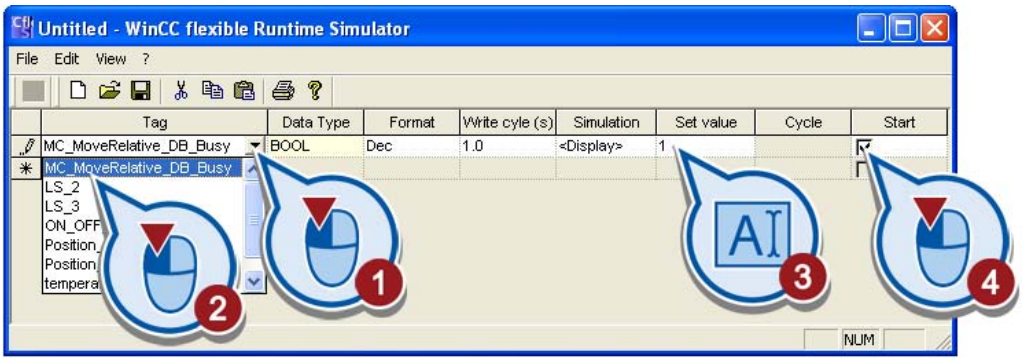
The simulation of the runtime is started. After the start of the simulation the HMI screen is displayed in the "RT Simulator" window and the red LED flashes (machine switched off).

3. Start the machine.



The green LED flashes. Nothing changes with regard to the visibility of the first bottle, as this animation now depends on the bit value of the "MC_MoveRelative_DB_Busy" parameter that was automatically stored as HMI tag.

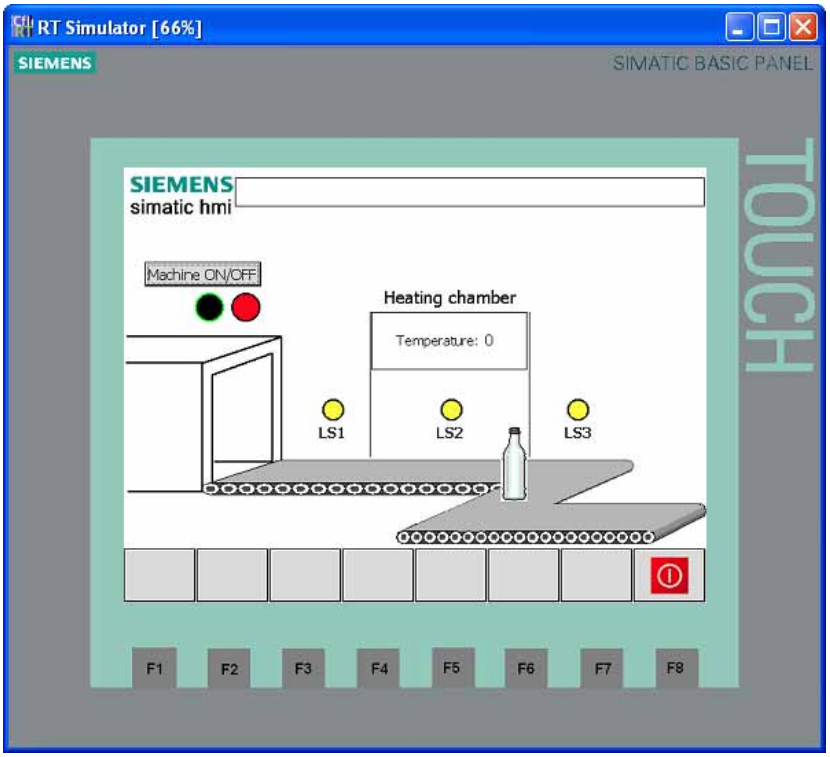
4. Set the value of the HMI tag "MC_MoveRelative_DB_Busy" to "1".



5. Switch to the "RT Simulator" window.

Result

The value for the "MC_MoveRelative_DB_Busy" tag is simulated. The bottle is moved on the conveyor relative to the position of the light barrier "LS3".



5.8 Start diagnostics view

Introduction

The following steps show you how to start the diagnostics view of the technological object "Axis". You can use the diagnostics function to monitor the motion jobs and the most important status and error messages of the motor axis.

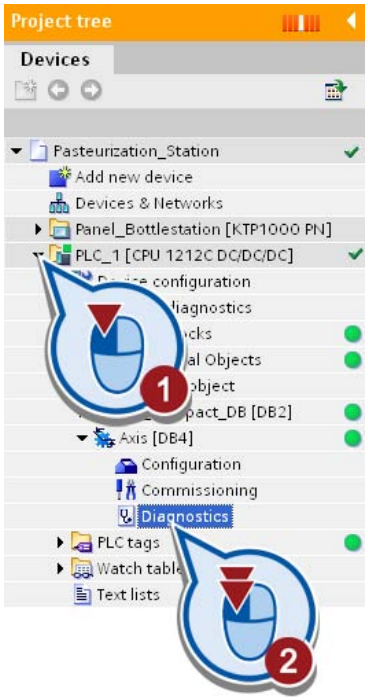
Requirements

- The technological object "Axis" has been created and correctly configured.
- The motion control instruction "MC_Power" has been created in the organization block "Main [OB1]".
- The motion control instruction "MC_MoveRelativ" has been created in the organization block "Main [OB1]".

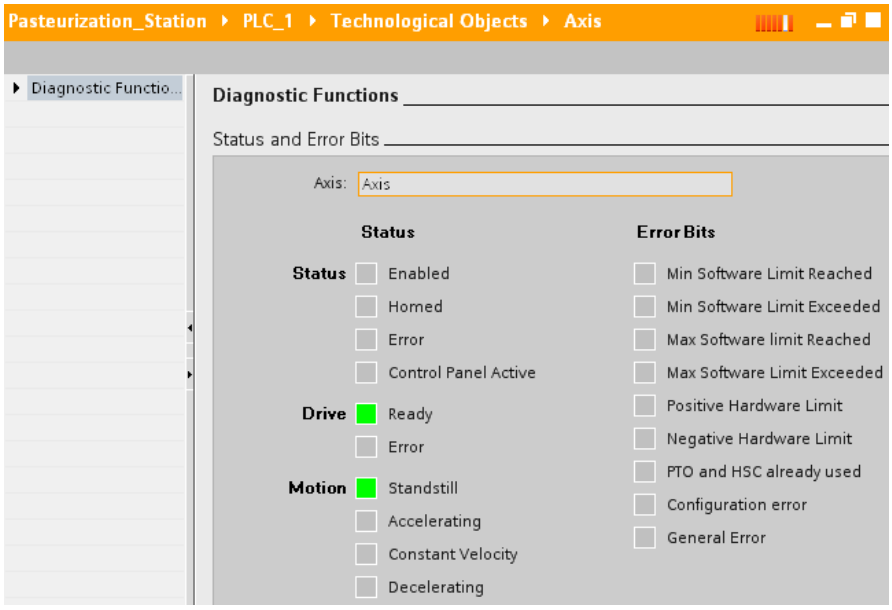
Procedure

To start the diagnostics function, follow these steps:

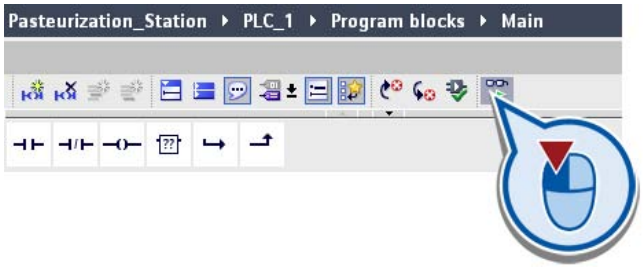
1. Load the program to the PLC and activate the online connection. For additional information see the "Loading the program to the target system (Page 48)" section.
2. Open the diagnostics window of the technological object "Axis".



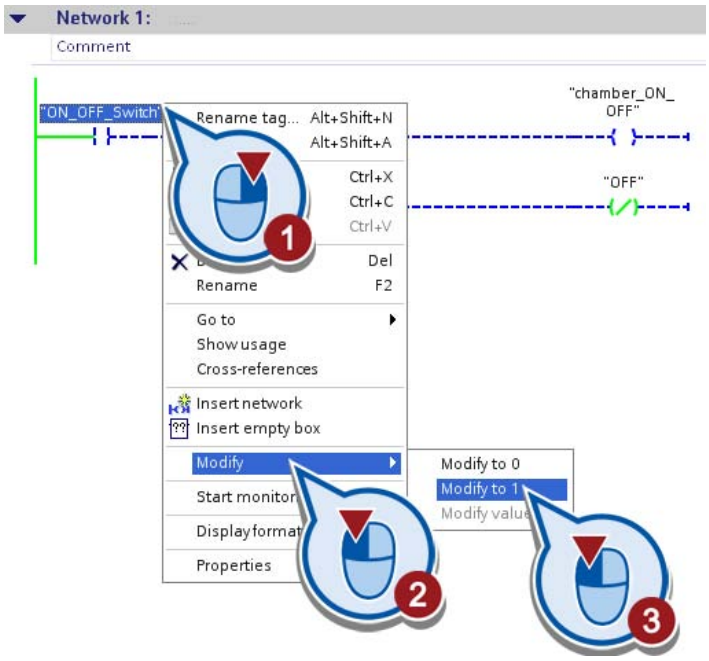
The diagnostics window opens.



3. Open the organization block "Main [OB1]".
4. Click the "Monitoring on/off" button on the toolbar of the program editor.



5. In Network 1, modify the "ON_OFF_Switch" tag to "1".

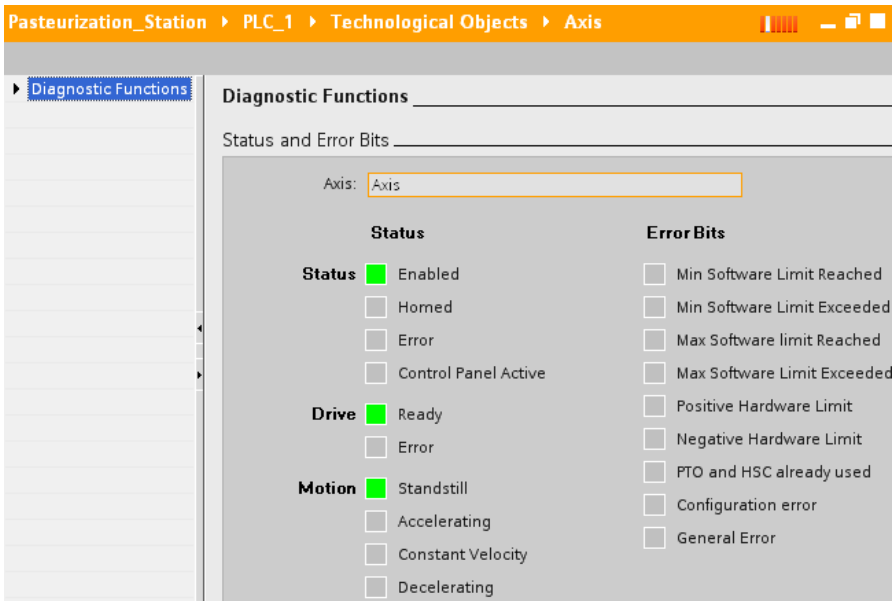


In network 1, the "ON_OFF_Switch" tag is set to the signal state "1". The current flows through the normally open contact to the coils at the end of the network. The "ON" tag is set and the example machine thereby switched on.

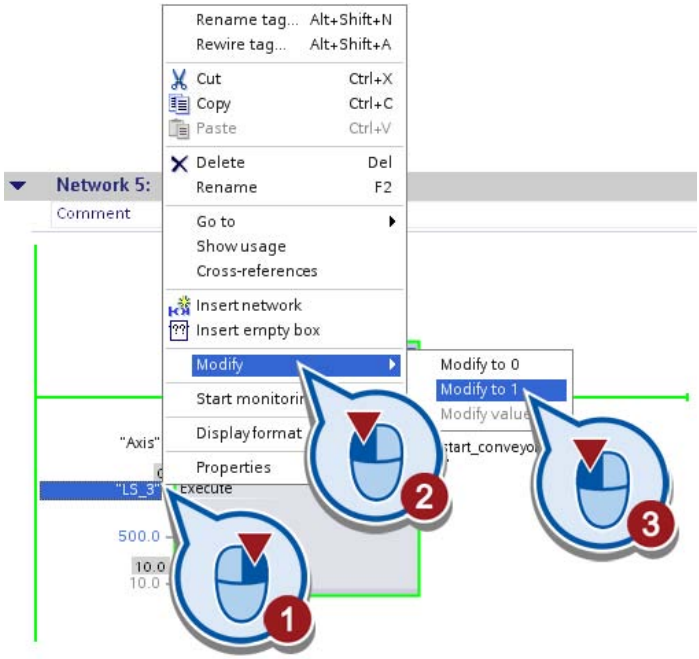
In network 4, the "MC_Power" instruction is executed and the motor axis of the conveyor enabled.

6. Switch to the diagnostics of the technological object "Axis".

   The enabling of the axis is displayed in the "Status" area of the diagnostics window.



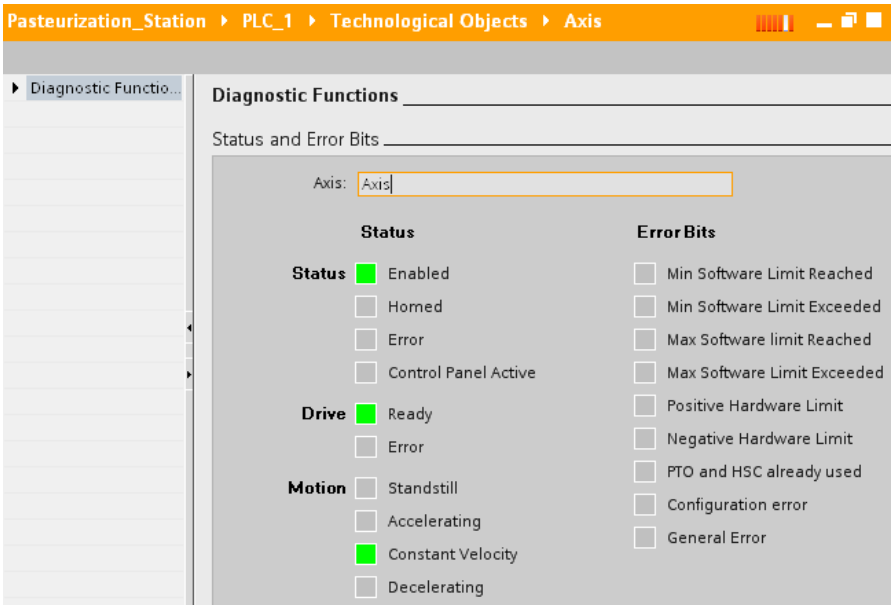
7. In Network 5, modify the "LS_3" tag to "1".



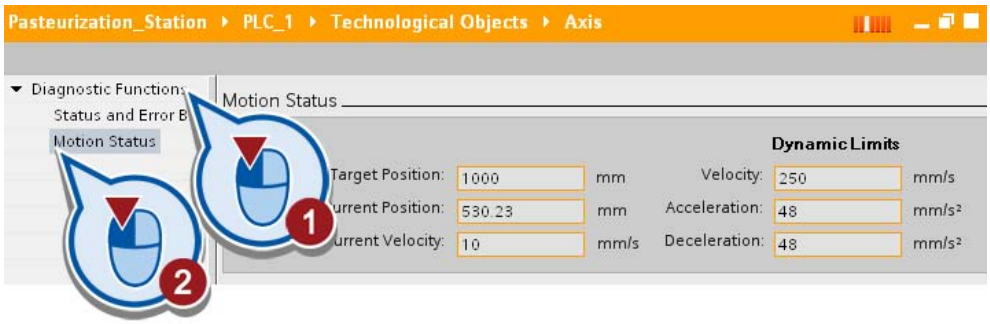
The "MC_MoveRelative" instruction is executed and the motor axis of the conveyor is set into motion.

8. Switch to the diagnostics of the technological object "Axis".

The "Motion" area of the diagnostics window shows that the motor axis is moving at a constant speed.



9. Display the motion status.



The current position of the motor axis is shown in the "Current position" field. When the target position of 500 mm relative to the start position is reached, the axis is set to stop.

Result

You have used the diagnostics function to check the proper functioning of the motor axis.

Glossary

Address

Designation of a certain address in the input, output or bit memory area of the CPU.

Addressing

Assignment of an address in the user program. Addresses can be assigned specific operands or ranges of operands. Examples: Input I12.1; memory word MW25.

Automation system

An automation system is a programmable logic controller (PLC) consisting of a central controller, a CPU, and various input/output modules.

Bit memory

Memory area in the system memory of a CPU. This can be written and read (in bits, bytes, words, and double words). The bit memory is available to the user to save interim results.

Block

Structures the user program in independent sections. Parts of the user program can be subdivided into blocks that can be reused at various locations or to make the structure of the user program more straightforward.

Block parameters

Placeholder within blocks that can be used more than once and that are supplied with current values when the relevant block is called.

Box

Boxes are program elements with complex functions. The empty box is an exception. You can use the empty box as a placeholder in which you can select the required operation.

Coil

You can use coils to modify binary operands. Coils can set or reset a binary operand depending on the signal state of the result of logic operation.

Configuring

"Configuring" is understood to mean arranging, setting and networking devices and modules within the device or network view. Racks are represented symbolically. Just like "real" racks, they allow you to plug in a defined number of modules.

Contact

You can use contacts to create or interrupt a current-carrying connection between two elements. The current is relayed from left to right. You can use contacts to query the signal state or the value of an operand and control it depending on the result of the current flow.

Control

A control is a process in which the result (output variable) influences a controlled variable by means of a feeback loop.

CPU

The user program is stored and executed in the central processing unit (CPU) of an automation system. It contains the operating system, processing unit, and communication interfaces.

CPU operating system

The operating system organizes all functions and sequences of the CPU that are not associated with a specific control task.

Cycle time

The cycle time is the time that the CPU requires to execute the user program once.

Cyclic interrupt

Cyclic interrupt OBs serve to start program in periodic time intervals independently of the cyclic program execution. The start times of a cyclic interrupt OB are specified using the time base and the phase offset.

Data block (DB)

Block in the user program that is used for storing values or character strings. There are global data blocks that can be accessed by all code blocks and instance data blocks that are assigned to a specific FB call.

Data type

Specifies how the value of a tag or constant is to be used in the user program. A tag of the BOOL data type can, for example, only take the value 1 or 0.

Function block (FB)

According to IEC 1131-3, a function block is a code block with static data. An FB allows you to pass parameters in the user program. This makes function blocks ideally suited for programming frequently recurring complex functions, such as closed-loop control or operating mode selection. Because a function block has a memory (instance data block), its parameters can be accessed at any time and at any point in the user program.

HMI device

Screen device for displaying the status, progress and operation of the user program.

I/O field

The I/O field is an input and output field that serves for displaying and changing the tag values.

Input

Memory area in the system memory of the CPU (process image input) or connection to an input module.

Instance data block

An instance data block stores the formal parameters and static data of function blocks. An instance data block can be assigned to an FB call or to a call hierarchy of function blocks.

Library

Collection of elements that can be used more than once.

Modify tags

Using the "modify tags" function, you can modify the tags of a user program and assign permanent values to individual tags at a specified point during execution of the user program.

Motion control

Software components for drive control. The motion control instructions control the technological object "Axis". The technological object "Axis" maps an axis in the PLC and is suited for activating stepper motors and servomotors with pulse interface.

Network

The program of a block is divided into networks. The networks are used to structure programs.

Organization block

Organization blocks (OBs) are the interface between the CPU operating system and the user program. The order in which the user program executes is specified in organization blocks.

Output

Memory area in the system memory of the CPU (process image of outputs) or connection to an output module.

PID controller

The PID controller continuously detects the measured actual value of the controlled variable within a control loop and compares this with the desired setpoint. The PID uses the resulting control deviation to calculate a controller output that adjusts the controlled variable (i.e. the variable that is to be controlled) as quickly and smoothly as possible to the setpoint.

PLC

Programmable logic controllers (PLCs) are electronic controllers whose functionality is stored as a program on the control device. The installation and wiring of the device do not therefore depend on the function of the PLC. A PLC is made up of at least one power supply module, one CPU, and input and output modules.

PLC tag table

A table used to define tags valid in the entire CPU.

Process image

The signal states of the digital input and output modules are stored on the CPU in a process image. A distinction is made between the process image input (PII) and the process image output (PIQ).

The process image output (PIQ) is transferred by the operating system to the output modules before the execution of the user program and before the reading of the process image input.

The process image input (PII) is read from the input modules by the operating system before the user program is started.

Program

A program solves a self-contained control task. It is assigned to a programmable module and can be structured in smaller units, for example blocks.

Programming device

A compact version of a personal computer designed specially for industrial use. A programming device (PG) is completely equipped for programming SIMATIC automation systems.

Programming language

A programming language is used to create user programs and provides a defined language subset for this purpose in the form of graphical or textual instructions. These instructions are entered by the user with an editor and then compiled into an executable user program.

PTO

Abbreviation for "Pulse Train Output". Some CPUs, such as the S7-1200, can generate quick pulse trains via the outputs, which are used to control the motion control operations.

Runtime

The runtime software executes the project in process mode and allows processes to be operated and monitored.

Subnet

A subnet includes all the network devices interconnected without gateways. It can include repeaters.

Tag

A tag is made up of an address and a symbolic name that is generally used multiple times in the project. The address (for example, of an input or bit memory) is used in communication with the automation system. Tags are used to perform a change of address (for an input, for example) centrally instead of throughout the entire user program.

Target system

Automation system on which the user program runs.

Watch table

The purpose of the watch table is to assemble the tags from the user program that are to be monitored, modified, and/or forced.